זאב נוימן
ישראל בורוביץ'

מחשבים ומערכות מידע

ניהול

۳۲۱۲
[illegible]

מחשבים ומערכות מידע

זאב נוימן, ישראל בורוביץ׳

הפאקולטה לניהול
בית-הספר למוסמכים במינהל עסקים ע"ש לאון רקנאטי
אוניברסיטת תל-אביב

מחשבים ומערכות מידע

זאב נוימן
פרופסור חבר

ישראל בורוביץ'
מרצה בכיר

"גומא" — ספרי מדע ומחקר
צ'ריקובר מוציאים לאור בע"מ
תל-אביב/תשל"ז

עיצוב העטיפה: שרה ברקאי

נדפס בישראל
סידור ועימוד: ש. אזולאי
דפוס חבצלת, הרצליה
1976

תוכן העניינים

הקדמה

המחשב ויישומיו משפיעים כיום ועתידים להשפיע על כל אורחות החיים. ההכרה בעובדה זו מוצאת לה ביטוי במידה רבה במערכות החינוך והניהול בישראל. רוב הפאקולטות במוסדות להשכלה גבוהה כוללות בין דרישותיהן שיעורי חובה בשטח המחשבים, וניצנים ראשונים של הוראה במקצוע זה מופיעים גם בבתי ספר תיכוניים. תוכניות לקידום מנהלים והשתלמויות בתחומי הניהול כוללות במידה גוברת והולכת נושאים שבהם עוסק ספר זה.

מחשבים ומערכות מידע בא לספק את הרקע העיוני להוראת נושא המחשב ופעולתו בארגונים. מטרת הספר להקנות לקורא בסיס להבנת הפעולה, המבנה והשימושים של המחשבים ומערכות המידע וכן העמדת טרמינולוגיה שתאפשר לו תקשורת טובה יותר עם מומחי המחשבים. מטרת הספר אינה ליצור מקוראיו מומחי מחשבים, מנתחי מערכות, או תוכניתנים.

מחשבים ומערכות מידע, המבוסס על נסיון ההוראה של המחברים בנושא זה בקורסים בסיסיים לתלמידי מינהל עסקים ובתוכניות שונות להכשרת וקידום מנהלים, עשוי לשמש ספר לימוד בנושא המחשב ומערכות מידע. היקף הספר נראה למחברים כמתאים לקורס סמסטריאלי העוסק בהוראת יסודות המחשב, התכנות, ומערכות המידע.

מחשבים ומערכות מידע מחולק לפרקים העוסקים בתיאור המחשב ורכיביו, תכנות, מערכות מידע והיבטים ניהוליים וכלכליים של מערכות ממוחשבות. לאחר הסבר קצר על דרך פעולתו של המחשב נסקרים הנושאים של תוכנה, תקשורת וארגון קבצים. המחברים בדעה שכל המשתמש במערכת מידע חייב להבין מה פירוש "לתכנת מחשב". מטעם זה נכלל בפרק התכנות גם לימוד של שפת התכנות BASIC. שפה זו נבחרה על שום היותה פשוטה, מצומצמת מבחינת כללי התחביר והדקדוק שלה, קלה מאד ללימוד ונוחה לשימוש במסופים. השימוש במסופים על־ידי סטודנטים, מדענים, חוקרים ומנהלים עתיד להתפשט, ומכאן החשיבות בידיעת שפה זו, שכיום הנה שפת התקשורת הנפוצה ביותר. הקורא אשר יש לו גישה למסוף המחובר למערכת מחשב המאפשרת שימוש בשפה זו, יוכל לתרגל בעצמו כתיבת תוכניות.

מערכות מידע ממוחשבות תהווינה חלק מרכזי בכל ארגון ולכן נכלל בספר פרק הדן בתכונות של מערכות אלו, בשלבי הפיתוח והאחזקה שלהן ובשילובן במערכת הניהול. המחשב, בצירוף ציוד העזר, מערכת התוכניות המפעילות אותו וכוח האדם הקשור בו הנו משאב יקר, לכן מסיים את הספר פרק הדן בהיבטים ניהוליים וכלכליים של מערכות ממוחשבות — ארגון וניהול מערכת ממוחשבת, הערכתה ותהליך בחירת המחשב.

לנוחיות הקורא צמוד ליד כל מונח מקצועי המופיע בספר בעברית גם תרגומו האנגלי. הקורא יכול להיעזר גם במילון המונחים האנגלי־עברי הקצר המופיע בסוף הספר.

חובה נעימה למחברים להודות למיכאל אדיב וליוסף מלמד, תלמידי הפאקולטה לניהול של אוניברסיטת תל־אביב, שתרמו להכנת הספר הן בתוכן והן בצורה.
תודה שלוחה גם לתהלה בלטר שהדפיסה בדייקנות ובנאמנות את הנסחים השונים של הספר.
ולבסוף, מיכה הדס היה בחינת אורים ותומים בנושא מערכות מידע. על תרומתו לגיבוש הנושא, שהיתה מכרעת, אנו חבים לו תודה מיוחדת.

המחברים

I. חלק ראשון

מחשבים -
מבנה
ודרך פעולה

.1

מהו מחשב?

מחשב (Computer)

המחשב הינו התקן אלקטרוני המסוגל לאחסן תוכנית שתפעיל אותו לביצוע סידרת פעולות אריתמטיות ולוגיות, וכן פעולות העברה ועריכה של נתונים.

מה עושים מחשבים אלקטרוניים ?

מחשבים "קוראים", "זוכרים", "כותבים" ומבצעים פעולות חשבון בהתאם להוראות התוכנית המפעילה אותם.
לדוגמא, מחשב יכול :

— לקרוא מאות כרטיסים בדקה.
— להדפיס מאות שורות בדקה.
— לאחסן מליוני אותיות או ספרות.
— לבצע חישוב מתימאטי מסובך מליוני פעמים ללא שגיאה.

השוואה בין אדם למחשב

	אדם	**מחשב**
כושר חשיבה	מותנה בכושר לימוד, אינטואיציה ואינטליגנציה	פועל רק בהתאם להוראות
		מידת הסיבוך אינה מפריעה
אגירה (זיכרון)	יכולת אגירה רבה — השליפה תלויה בכושר הזיכרון ולכן הנה סלקטיבית	יכולת אגירה גדולה אולם מוגבלת — השליפה מדויקת
מהירות	איטית	מהירות אלקטרונית גבוהה
דיוק בביצוע חישובים ופעולות	בינוני — בלתי יציב	מוחלט
כושר עבודה	מוגבל	מכונה
נקיטת עמדה	סובייקטיבי	אובייקטיבי — נייטרלי

דוגמאות של שימושים אופיניים למחשבים

— הנחיית טילים ומטוסים.
— הכנת המחאות לתשלום.
— תכנון מיני דלק שונים.
— חיזוי מזג אוויר.

בכל השימושים האלו, המחשב:
— מקבל נתונים ממקור כלשהו.
— מאחסן נתונים אלו ממערכות הזיכרון שלו.
— מעבד את הנתונים כפי שמכתיבה לו תוכנית פעולתו (שנכתבה על־ידי אדם).
— מייצר את התוצאות המבוקשות בצורה הרצויה.

סוגי מחשבים

מחשבים מתחלקים לשני סוגים עיקריים:
— מחשב אנאלוגי (Analog Computer).
— מחשב סיפרתי (Digital Computer).

א. מחשב אנאלוגי

מחשב אנאלוגי מקבל קלט בצורת מתחים חשמליים שונים המבטאים מדידות של תופעות פיסיקליות. המחשב מעבד אינפורמציה זו, ומוציא פלט באמצעות מכוונים, מסכים, גרפים או אותות חשמליים המשמשים לבקרת תהליכים, לוויסות לחצים וכדומה. מד הדלק של מכונית הוא דוגמא לכלי אנאלוגי.
כתוצאה משיטת עבודתו, למחשב האנאלוגי דיוק המוגבל על־ידי דיוקם של מכשירי המדידה והציוד.
שימושו העיקרי של מחשב אנאלוגי הוא בתהליכי ייצור ובקרה.

ב. מחשב סיפרתי

מחשב סיפרתי הנו מכשיר המבצע חישובים באמצעות מספרים המבטאים ערכי משתנים וכמויות. החישובים המספריים מאפשרים להגדיר כמויות באופן סיפרתי בכל מידת דיוק מבוקשת.
המחשבים המשמשים לעיבוד נתונים אלקטרוני (ענ"א) הנם סיפרתיים.
ספר זה עוסק במחשבים סיפרתיים בלבד.

דורות מחשבים

מבחינת ההתפתחויות הטכנולוגיות נהוג לציין כמה דורות של מחשבים:
1. הדור הראשון, שהחל ב־1952 ונמשך שש שנים.

מאפייניו העיקריים:

א. שימוש בשפופרות רדיו.

ב. זיכרון טבעות ושימוש בתוף מגנטי לזיכרון פנימי.

ג. עקרון הגישה הישירה — זיכרון חיצוני של תקליטים מגנטיים.

2. הדור השני, בשנים 1958—1963, תקופה של שש שנים בדומה לדור הראשון.
מאפייניו העיקריים:

א. שימוש בטרנזיסטורים ומוליכים מוצקים, שהקטין את מימדי המחשב, הגביר את יעילותו והקטין את הוצאות השימוש.

ב. התחלת השימוש בסרטים מגנטיים, שאיפשר אחסון חיצוני נרחב וזול יחסית.

ג. פיתוח שיטות זמן אמיתי (real-time) ושיתוף זמנים (time-sharing).

ד. התגבשות תפישת ה"מודולוריות", שאיפשרה הרחבת מערכת קיימת במקום החלפתה במערכת חדשה.

3. הדור השלישי, בשנים 1964—1969, תקופה של שש שנים.
מאפייניו העיקריים:

א. שימוש במעגלים משולבים שהקטין את זמן החישוב, הגדיל את האמינות, והקטין את מימדי המחשב.

ב. יכולת עיבוד בסביבה רב-תוכניתית (multiprogramming).

ג. יכולת עיבוד בו-זמנית (multiprocessing).

4. הדור הנוכחי, החל משנת 1970.
מאפייניו העיקריים:

א. שימוש במעגלים משולבים מיקרוסקופיים, שאיפשר צמצום נפח וצריכת החשמל ב-50% (Monolithic Systems Technology, Large Scale Integration).

ב. שיפורים בקיבולת הזיכרון באמצעות זיכרון לייזר (Laser Mass Memory).

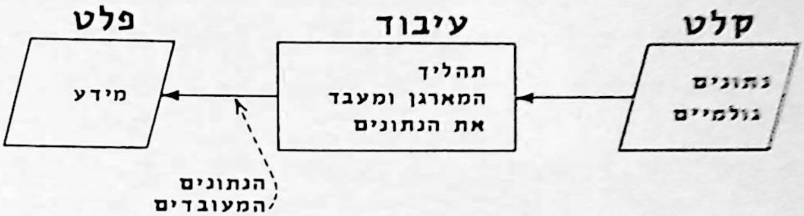
בנוסף להתפתחויות טכנולוגיות אלו פותחה בשנים אלו תפישה של מרכז מחשבים המספק מיגוון רחב של שירותי מחשב, היכולים לענות על כל דרישות עיבוד נתונים אוטומטי של ארגון (Computer Utility Concept).

נתונים, עיבוד נתונים ומידע

נתונים (Data) הנם עובדות גולמיות בלתי מעובדות.

מידע (Information) פירושו נתונים מעובדים ומאורגנים בצורה משמעותית לגבי פתרון בעיה מסוימת.

עיבוד נתונים (Data Processing) הוא התהליך המקבל נתונים כקלט, מארגנם, מעבדם, ומוציאם כמידע — הפלט של התהליך (ראה תרשים 1.1).



תרשים 1.1: תהליך עיבוד הנתונים

כדוגמא להפקת מידע מנתונים נתבונן בתהליך מציאת הכנסות של חברה מתוך נתוני מכירה שוטפים (תרשים 1.2):



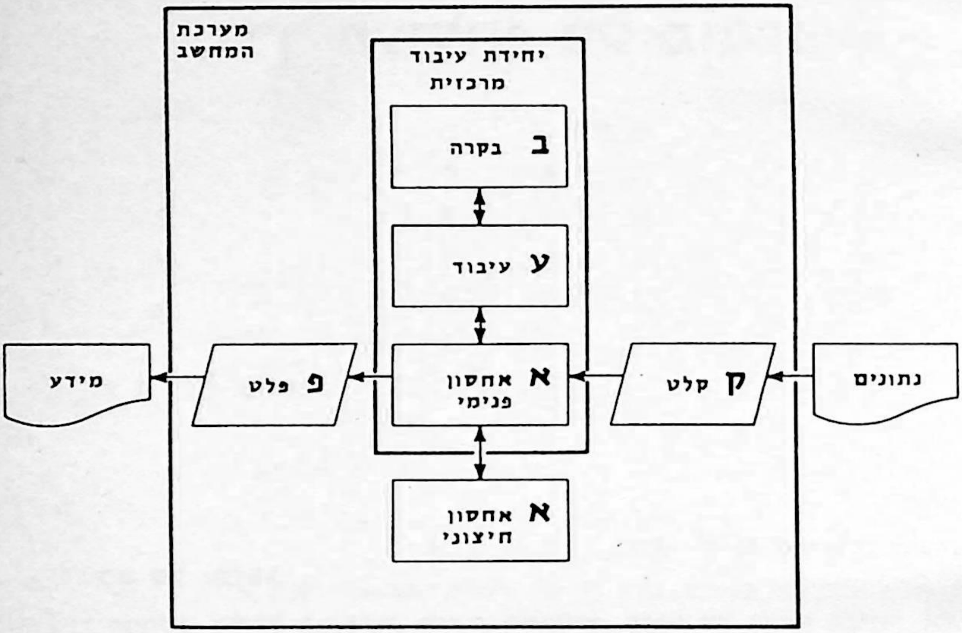
תרשים 1.2:

תהליך "קאעבפ"

את האלמנטים היסודיים של תהליך עיבוד הנתונים ניתן לסכם כך (תרשים 1.3):

1. **קלט** (Input) — הנתונים נקראים באמצעות התקנים שונים כגון קוראת-כרטיסים, לוח-מקשים.
2. **אחסון** (Storage) — הנתונים שנקראו מאוחסנים באחת משתי דרכים אפשריות:
   א. בזיכרון פנימי המשמש לאחסון הוראות ונתונים במהירות גישה גבוהה.
   ב. בזיכרון משני (חיצוני) המשמש לאחסון נתונים במהירות גישה איטית.
3. **עיבוד** (Processing) — עיבוד הנתונים מתבצע ביחידה האריתמטית-לוגית של המחשב.
4. **בקרה** (Control) — הבקרה מתבצעת באמצעות יחידת הבקרה והפיקוח של המחשב.
5. **פלט** (Output) — המידע מופק באמצעות התקנים שונים, כגון מדפסות, מסכים וכו'.

פונקציות אלו קיימות בכל תהליך עיבוד נתונים, בין אם במחשב ובין אם במערכת כרטיסים מנוקבת, במערכת ידנית, או במערכת הפשוטה ביותר של עפרון ונייר.



תרשים 1.3: תהליך קאעב"פ

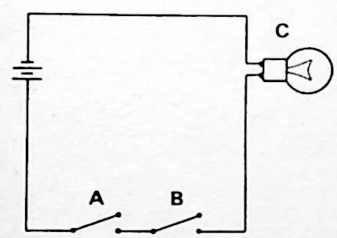
.2
דרך פעולתו של מחשב

לוגיקה של מחשב

ציוד המחשב, הפועל באמצעות אותות חשמליים, מנצל את הצופן הבינרי (להלן) תוך שימוש ביסודות האלגברה הבוליאנית.
טענה בוליאנית יכולה להיות נכונה (TRUE) או בלתי נכונה (FALSE), דבר הניתן לביטוי באמצעות כל מכשיר חשמלי בעל 2 מצבים, כגון נורה חשמלית.
ניתן לעבד טענות אלה באמצעות פעולות AND, OR, NOT המוגדרות כך:

AND

המצב C יהיה "נכון" כאשר A וגם B "נכונים", היינו שני המתגים סגורים (תרשים 1.4).



תרשים 1.4: A AND B

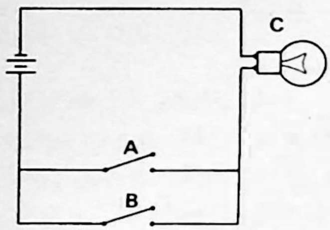
A AND B

A	B	C
0	0	0
0	1	0
1	0	0
1	1	1

"1" מייצג "נכון", או מתג סגור, או זרימת זרם.
"0 מייצג "לא-נכון", או מתג פתוח, או אי-זרימת זרם.

OR

המצב C יהיה "נכון" אם A "נכון" או B "נכון", או שניהם "נכונים".
ניתן לסכם יחס זה בדיאגרמת מעגלים חשמליים: C "נכון" כאשר **הנורה דולקת**, A ו־B מצויינים על־ידי מתגים (תרשים 1.5).



A OR B

A	B	C
0	0	0
0	1	1
1	0	1
1	1	1

תרשים 1.5: A OR B

NOT

פעולת NOT משמעותה הפיכת 1 ל־0 או 0 ל־1.
במחשב מבוצעות פעולות אלה על־ידי שערים (Gates) OR, AND ו־NOT.

שערים AND ו־OR (תרשים 1.6)

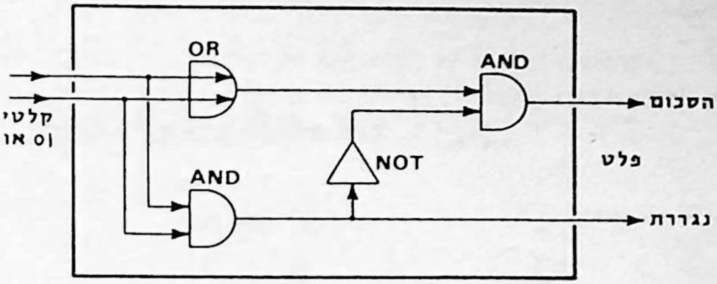
שער AND יוצר 1 רק כאשר שני הקלטים שווים ל־1, אחרת יוצר 0.
שער OR יוצר 1 כאשר אחד הקלטים או שניהם שווים ל־1, אחרת יוצר 0.



תרשים 1.6:

הפעולות הבסיסיות שמחשב מסוגל לבצע הן העברה, חיבור והשוואה של נתונים. פעולות אלה נראות, לכאורה, בלתי־מספיקות לביצוע עיבודים מורכבים, אך באמצעות חזרות רבות, מהירות גבוהה ותיחכום בשימוש בפעולות הבסיסיות, מבצע המחשב את העיבודים.

מחבר למחצה (Half Adder) — פעולת חיבור של שתי ספרות, 0 או 1 מתבצעת על־ידי שימוש בפעולות AND, OR, NOT. הפלט המתקבל ממחבר למחצה יכול להיות תוצאה סופית של סיפרה אחת, או תוצאה של סיפרה ונגררת (תרשים 1.7).



תרשים 1.7: מחבר למחצה

1 + 1 ← 10 (0 = הסכום, 1 = הנגררת)

0 + 1 ← 1

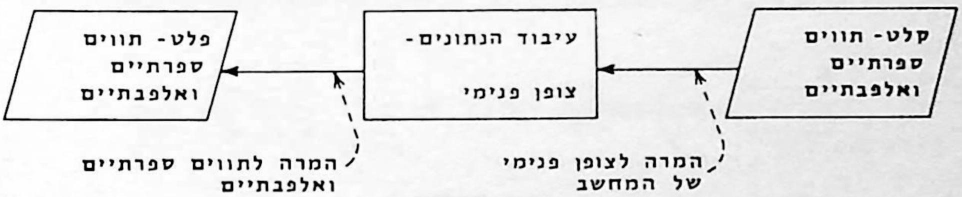
0 + 0 ← 0

מחבר מלא (Full Adder)

מורכב ממחברים למחצה ומבצע פעולות חיבור לכל מספר המורכב מספרות 0 ו־1.

פעולת המחשב

נתוני קלט ופלט במערכת מחשב מוצגים בשיטת המספרים העשרונים (decimal code) ובתווים אלפבתיים. לעומת זאת הפעולות הפנימיות במחשב נעשות בצופן שונה, בדרך כלל בצופן הבינרי (binary code).



תרשים 1.8: המרה מצופן לצופן במחשב

כדי להבין את פעולתו הפנימית של המחשב יוגדרו תחילה מספר מונחים בסיסיים:

הסיבית (תג, Bit)

הסיבית (ראשי תיבות של "סיפרה בינרית") הנה המרכיב היסודי של כל מערכת מחשב סיפרתי. היא מייצגת "נתח" בודד של מידע. לדוגמא: נוכחותו או העדרו של חפץ במקום מסוים. המידע מיוצג על־ידי הימצאות הסיבית באחד מתוך שני מצביה האפשריים בלבד.

תו (Character)

התו מורכב מסיבית אחת, או מסיביות אחדות. לתו צורות אפשריות רבות, שלא כסיבית המוגבלת לשתי צורות בלבד. תו יכול להיות סיפרה, אות מאותיות האלף־בית, או סימן מיוחד, כמו רווח (space), סימן ניקוד, או סימן אלגבראי.

תו מורכב לרוב מארבע סיביות. לעתים קרובות משתמשים **בתו** **וסיפרה** כמלים נרדפות.

לדוגמא: המספר 7 הוא תו (סיפרה), בעוד שהמספר 17 הנו מספר המורכב משני תווים.

מלה (Word) של מחשב היא צירוף של תווים. המלה 3456 מכילה ארבעה תווים ונקראת **נומרית**. המלה AX56L8 מכילה ששה תווים. מילה כזאת נקראת **אלפא־נומרית**, מכיוון שיש בה גם מספרים וגם אותיות אלפביתיות. המלה ACDB מכילה ארבעה תווים ונקראת **אלפביתית**, מכיוון שיש בה רק אותיות אלפבתיות.

הסב־סוב — היחידה היסודית של המחשב

מחשבים מעבדים אינפורמציה סיפרתית על־ידי טיפול בקבוצות של סיביות בסיסיות. במחשבים סיפרתיים ישנם מעגלים אלקטרוניים הידועים כמעגלי סב־סוב (flip-flop), שלהם שני מצבים אפשריים. הסיבית מיוצגת באמצעות מצבי הסב־סוב.

לדוגמא: נורה — אם היא "דולקת" היא מציינת "לילה"
אם היא "כבויה" היא מציינת "יום"

ואז חלק היממה מוגדר לחלוטין.

שים לב — מתוארים רק שני מצבים; למשל, שעת היום אינה מצויינת.

הצופן הבינרי

הגדלת מספר המצבים הניתנים להצגה בצופן הבינרי נעשית על־ידי הוספת סיביות. למשל, שימוש ב־2 נורות מאפשר ציון של 4 מצבים.

נורה 1	נורה 2
כבויה	דולקת
דולקת	כבויה
כבויה	כבויה
דולקת	דולקת

כך אפשר לתאר, למשל, בוקר, אחר־צהרים, ערב, לילה.

משתמשים ב־1 ו־0 כ"שמות" נוחים למצבים מנוגדים הניתנים להבחנה (on ו־off), כאשר את האפס מיחסים למצב הפחות פעיל של המכשיר (הנורה — במצב כבוי).

היחס בין מספר המכשירים הדו־מצביים, או הסיביות, לבין מספר התנאים השונים —

הוא יחס מעריכי (exponential). פירוש הדבר, ש־2 בחזקת מספר הסיביות שבשימוש, קובע את מספר האפשרויות. באופן זה:

עם 2 סיביות מספר המצבים הוא	$4 = 2^2$
3 " " " " "	$8 = 2^3$
4 " " " " "	$16 = 2^4$
5 " " " " "	$32 = 2^5$
10 " " " " "	$1024 = 2^{10}$
20 " " " " "	$1{,}048{,}576 = 2^{20}$

יחס זה הוא הבסיס לצופן הבינרי.

אם נשתמש ב־10 האצבעות של כפות ידינו, הרי שבשיטה העשרונית אפשר לספור עד 10 בלבד. אבל אם בשיטה הבינרית ניתן לאצבע הראשונה את הערך "1" ולאצבע השנייה כפליים — "2", ולשלישית כפול מהשנייה — "4", וכך הלאה, נראה שבעשר אצבעות נוכל לספור עד 1024 (2^{10}).

שים לב — סך־הכל של האצבעות הוא 1023, אבל "0" נחשב גם כן.

אצבע:	1	2	3	4	5	6	7	8	9	10
ערך:	1	2	4	8	16	32	64	128	256	512

למשל:

את המספר 3 אפשר לסמן על־ידי 2 אצבעות בלבד: 1 ו־2
את המספר 4 אפשר לסמן על־ידי 1 אצבע בלבד: 3
את המספר 5 אפשר לסמן על־ידי 2 אצבעות בלבד: 1 ו־3
את המספר 17 אפשר לסמן על־ידי 2 אצבעות בלבד: 1 ו־5

צפנים מצורפים

הפגם בשיטה הבינרית הוא שמעט מאד אנשים חושבים במונחים של מספרים בינריים בסיסיים. לדוגמא, קשה מאד להבחין בהבדלים שבין שני המספרים הבינריים הבאים:

1 1 1 1 1 1 1 1 1 0 = 1023 (הערך העשרוני)
1 1 1 1 1 1 1 1 1 1 = 2047 (הערך העשרוני)

ישנם צפנים רבים המורכבים מהצופן הבינרי הבסיסי. השימוש בהם נובע מהמבנה הפנימי של המחשב. למשל, הצופן העשרוני על בסיס בינרי (binary coded decimal code), שהוא צירוף של השיטה הבינרית והשיטה העשרונית.

כמו כן קיימים צפנים אחרים שיתוארו להלן:
הצופן האוקטלי
צופן עודף 3
הצופן ההקסאדצימלי

צופן עשרוני על בסיס בינרי (Binary Coded Decimal)

לצופן זה יתרונות שימושיים רבים במחשבים. הספרות העשרוניות מ־0 עד 9 ניתנות להגדרה על־ידי ארבע סיביות בינריות.

בינרי	עשרוני
0000	0
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7
1000	8
1001	9

בכל מספר בינרי — הטור הימני הוא טור היחידה 1 $(2^0 = 1)$
הטור השני מימין מייצג 2 $(2^1 = 2)$
הטור השלישי מימין מייצג 4 $(2^2 = 4)$
הטור הרביעי מימין מייצג 8 $(2^3 = 8)$
באופן כזה המספר העשרוני 7 מורכב מ־ $4+2+1 = 7$
והמספר העשרוני 9 מורכב מ־ $1+8 = 9$

השיטה העשרונית על בסיס בינרי מבזבזת כמה אפשרויות, מכיוון שעל־ידי ארבע סיביות אפשר היה לייצג גם את המספרים העשרוניים: 10, 11, 12, 13, 14 ו־15.
כדי לייצג מספר עשרוני, המורכב מיותר מסיפרה עשרונית אחת, משתמשים בקבוצה נפרדת של 4 סיביות בינריות, בשביל כל סיפרה עשרונית.

לדוגמא: המספר 34 מיוצג בצופן עשרוני על בסיס בינרי כך:

0100	0011
4	3

נוח יותר להסתכל על זה בצורה אנכית.

לדוגמא: המספר 3586:

	6	8	5	3
	יחידות	עשרות	מאות	אלפים
$1 = 2^0$	0	0	1	1
$2 = 2^1$	1	0	0	1
$4 = 2^2$	1	0	1	0
$8 = 2^3$	0	1	0	0

לכן, המספר העשרוני 3586 מיוצג בצופן העשרוני על בסיס בינרי על ידי 0110 1000 0101 0011.
אפשר לראות שההפיכה לסימון עשרוני ישיר קלה יחסית.

חשיבות (Significance)
הסיביות להן שייכות החזקות הגבוהות יותר, חשובות יותר מאשר הסיביות להן שייכות החזקות הנמוכות.

המרה מצופן לצופן
מספר בינרי למספר עשרוני: מחברים את המשקלים של הסיביות שיש בהן 1.
דוגמא: המרת 10101 מבינרי לעשרוני:

(משקל)	2^0	2^1	2^2	2^3	2^4
(מספר בינרי)	1	0	1	0	1

משקל ספרה

$$1 = 2^0 \times 1 =$$
$$+$$
$$0 = 2^1 \times 0 =$$
$$+$$
$$4 = 2^2 \times 1 =$$
$$+$$
$$0 = 2^3 \times 0 =$$
$$+$$
$$16 = 2^4 \times 1 =$$

מספר עשרוני 21

מספר עשרוני למספר בינרי:
המרה זאת נעשית על ידי החסרת החזקה הגבוהה של שניים (2) האפשרית מהמספר העשרוני. במידה וניתן לחסר נכתוב "1" במקום המשוקלל המתאים. במידה ואחר החיסור הנכון לא ניתן לחסר את החזקה הבאה של 2

מהשארית שנותרה נכתוב "0". תהליך זה חוזר לגבי השארית ומסתיים כאשר השארית היא 0.

דוגמא: המרת 21 מעשרוני לבינרי.

החזקה של 2 הגדולה ביותר (אשר קטנה מ־21) היא 16 (2^4).

נחסיר מהמספר את החזקה שנתקבלה

$21 - 16 = 5$ 1

החזקה הבאה של 2 היא 8, אותו לא ניתן לחסר

מהמספר 5 ולכן יש לרשום 0 בסיבית המתאימה לו 10

החזקה הבאה היא 4

$5 - 4 = 1$ 101

החזקה הבאה היא 2, הגדול מ־1 1010

הסיפרה הבאה היא 1, המתאים לחזקה הנמוכה של 10101

2 (2^0). לכן נרשום 1. השארית היא 0 ולכן התהליך הסתיים.

מחשבים שונים משתמשים בצפנים השונים מהצופן הבינרי לייצוג פנימי של התווים:

צופן אוקטלי (Octal)

שיטה זאת מבוססת על בסיס 8. ייצוג הספרות 0—7 נעשה באותה דרך כפי שזה נעשה בשיטה העשרונית. שיטה זאת מכילה רק 8 ספרות.

בסיס 8 ← 0, 1, 2, 3, 4, 5, 6, 7

הסיפרה הגדולה ביותר בשיטה זו הנה 7. הסיפרה העשרונית 8 תיוצג בשיטה זו על־ידי המספר האוקטלי 10.

המרה מצופן בינרי לצופן אוקטלי נעשית על ידי חלוקת המספר הבינרי לקבוצות של 3 סיביות כל אחת והמרת הצופן הבינרי בכל שלשה לסיפרה עשרונית.

דוגמא: המרת המספר הבינרי 101001010 למספר אוקטלי נעשית כך:

נחלק את המספר לקבוצות של 3 סיביות מימין לשמאל ואחר־כך נחשב את הערך של כל שלשה בצופן אוקטלי:

בינרי	010	001	101
אוקטלי	2	1	5

ונקבל את המספר האוקטלי 512.

צופן הקסאדצימלי (Hexadecimal)

שיטה זאת מבוססת על בסיס 16. ייצוג הספרות 0—9 נעשה באותה דרך כפי שזה נעשה בשיטה העשרונית. אולם, כדי לייצג את הערכים העשרוניים 10—15

משתמשים באותיות A—F. לדוגמא, הערך העשרוני 10 מיוצג בשיטה זאת על־ידי האות A, 11 על ידי האות B וכך הלאה. שיטה זאת משתמשת ב־16 סימונים.

0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F → בסיס 16

המרה מצופן בינרי לצופן הקסאדצימלי נעשית בדומה להמרה מצופן בינרי לצופן אוקטלי:

מחלקים את המספר הבינרי לקבוצות בנות 4 סיביות כל אחת כמוצג בדוגמא הבאה:

בינרי	0111	0110	1100	0101	1111
עשרוני	7	6	12	5	15
הקסאדצימלי	7	6	C	5	F

ונתקבל המספר ההקסאדצימלי F5C67.

צופן עודף—3 (Excess—3 Code)

בחישובים פנימיים נעשה שימוש בשיטה זו כפי שנראה להלן:

לצורך חלוקת מספרים על־ידי חיבור, יש להפחית את הספרות מ־9. אפשר לעשות זאת מיד, אם הספירה מבוטאת בצופן עודף—3. כל מה שיש לעשות הוא להפוך את תוכנה של כל סיבית.

כלומר: מ־0 ל־1

או מ־1 ל־0

פעולה זאת נקראת השלמה (complementing).

השימוש בצופן עודף—3 דומה מאד לסימול מספרים בצופן עשרוני על בסיס בינרי. ההבדל ביניהם הוא שכל מספר בצופן עודף—3 גדול ב־3, מהמספר המקביל לו בצופן העשרוני על בסיס בינרי.

המשלים של עודף—3	עודף—3	בינרי	עשרוני
1100	0011	0000	0
1011	0100	0001	1
1010	0101	0010	2
1001	0110	0011	3
1000	0111	0100	4
0111	1000	0101	5
0110	1001	0110	6

צופן אלפאנומרי (Alphanumeric Code)

במחשבים המיועדים לשימושים מסחריים, יש צורך גם בתווים אלפאנומריים.

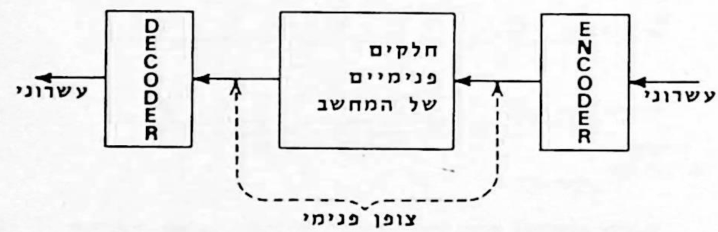
לדוגמא: בחישוב משכורת יש לאחסן במחשב גם את שם העובד ומידע אלפביתי אחר יחד עם הנתונים המספריים השייכים לו, כדי לקבל את הפלט הרצוי. תווים אלה אינם משמשים בחישובים.

ראינו כי באמצעות ארבע סיביות אנו מגדירים בשיטה העשרונית לפי צופן בינרי עשר ספרות ואיננו משתמשים בחמש אפשרויות נוספות לייצוג ספרות.

באמצעות שש סיביות אפשר להגדיר 64 צירופים שונים. לאחר שימוש בעשרה צירופים לייצוג הספרות, ישארו 54 צירופים עבור אותיות וסימנים שונים.

כיצד ממיר המחשב מצופן אחד למשנהו

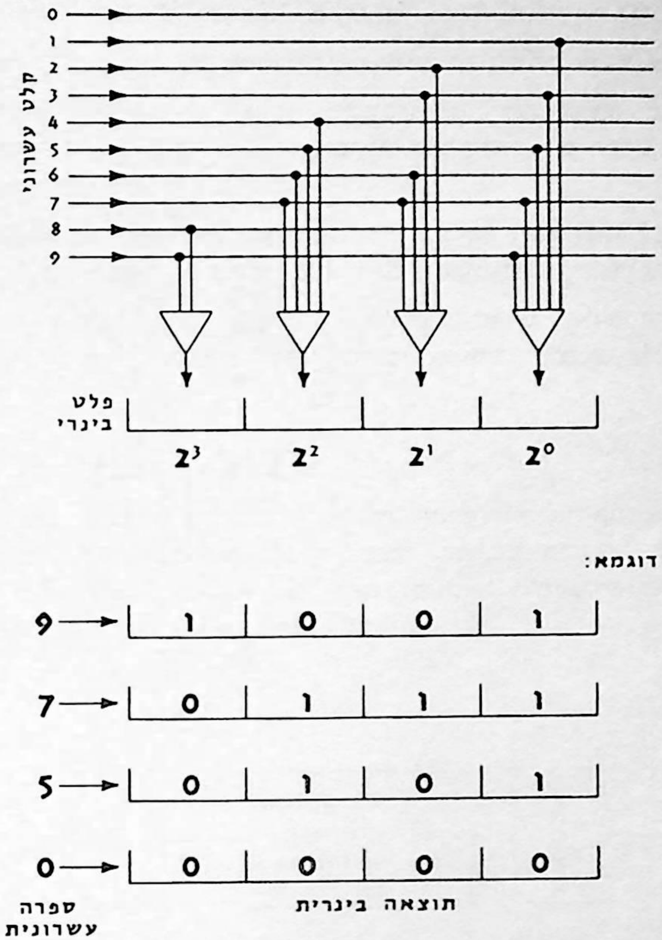
מכיוון שמחשבים עובדים בדרך כלל בשיטה האוקטלית או ההקסאדצימלית יש צורך להמיר מהשיטה העשרונית לשיטה שבה משתמש המחשב. המחשב מבצע זאת על ידי מכשיר הנקרא encoder. לאחר העיבוד מתורגמים המספרים חזרה למספרים עשרוניים על־ידי ה־decoder (תרשים 1.9).



תרשים 1.9: המרה מצופן לצופן

תרגום מעשרוני לבינרי (Decimal—to—Binary—Encoder)

התרשים הניתן להלן מציג המרה של סיפרה עשרונית למספר בינרי. לדוגמא, אם הסיפרה העשרונית 9 תוזן ל־encoder התוצאה תהיה 1001. ניתן לראות בתרשים 1.10 שבאותם תאים שאליהם מחובר הקלט תווצר הסיפרה 1, ובתאים הנותרים הסיפרה 0.



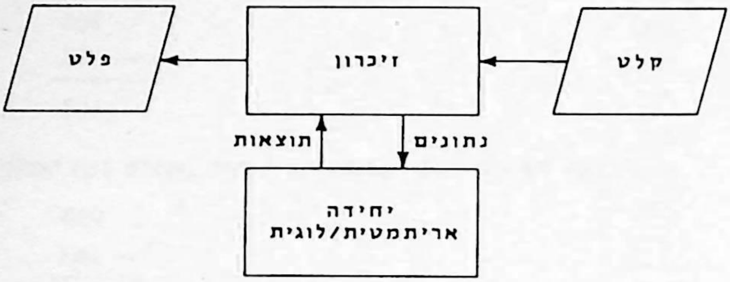
תרשים 1.10: תרגום סיפרה עשרונית למספר בינרי

הקורא יכול לבחון את פעולת ה־encoder על־ידי שימוש בטבלה הבאה:

Decimal	Binary	Decimal	Binary
0	0000	5	0101
1	0001	6	0110
2	0010	7	0111
3	0011	8	1000
4	0100	9	1001

דרכי החישוב של המחשב

המחשב מעבד נתונים על־ידי אחסנת הנתונים ושליפת חלקים נדרשים של נתונים אלה לחישוב. החישוב עצמו נעשה ביחידה האריתמטית/לוגית (תרשים 1.11). יחידה זו מסוגלת לבצע פעולות חיבור בלבד. כל שאר הפעולות האריתמטיות מבוצעות באמצעות פעולת החיבור.



תרשים 1.11: דרך החישוב של המחשב

פעולת החיבור במחשב

דוגמא: פעולת החיבור של המספרים 163 ו־265 תתבצע בשלושה צעדים:

	1	6	3
	+		
	2	6	5
צעד א'			8
		2(1)	
צעד ב'	+ 1		
צעד ג'	4		
	4	2	8

חיסור על־ידי חיבור

קל לחסר מספר ממספר שכל אחת מסיפרותיו היא 9. לדוגמא, קל להפחית 163 מ־999. התוצאה, 836, היא המשלים ל־9 של המחסר 163.

לצורך חיסור על־ידי חיבור נבצע 4 שלבים:

1. חשב את המשלים ל־9 של המחסר.
2. חבר למחוסר את המספר המשלים שהתקבל.

3. הוסף לתוצאה 1.
4. בסכום שהתקבל הזנח את הסיפרה השמאלית (החשובה), אשר תהיה תמיד 1.

התוצאה תהיה ההפרש המבוקש.

דוגמא:

```
  265
— 163
-----
  102
```

כדי לחסר 163 מ־265, נחשב את המספר המשלים של 163:

```
  999
— 163
-----
  836
```

נוסיף למחוסר את התוצאה:

```
  265
+ 836
-----
 1101
```

חיבור של 1 והזנחת הסיפרה השמאלית נותנים לנו את המספר 102, שהיא התשובה הנכונה.

כפל באמצעות חיבור

דוגמא:

```
   265
  ×163
------
   795
  1590
  265
------
 43195
```

למעשה, אנחנו אומרים: 3 פעמים 265 = 795, מזיזים טור אחד שמאלה, 6 פעמים 265 = 1590, מזיזים טור נוסף שמאלה, ופעם אחת 265.

לכן אפשר לבצע זאת בצורה הבאה:

```
  265
× 163
─────
  265
+ 265
+ 265
+ 265
+ 265
+ 265
+ 265
+ 265
+ 265
  265
─────
43195
```

ההכפלה במחשב נעשית בצורה זאת: המספר המוכפל פשוט מחובר לעצמו כמה פעמים, כמספר המופיע בסיפרת היחידות של הכופל. המחשב זז לאחר מכן טור אחד שמאלה, וכן הלאה.

חילוק באמצעות חיבור

גם חילוק אפשר לבצע באמצעות חיבור.

דוגמא: נניח כי יש לחלק את המספר 33 ב־8. נחסר 8 מ־33 ונבחן אם תוצאת הביניים היא חיובית או אפס. אם היא חיובית, אנו מוסיפים 1 למונה התוצאה (מונה זה הוא מקום מסוים במחשב, שנקבע לצבירת התוצאות). תהליך זה חוזר עד אשר מתקבלת תוצאה חיובית הקטנה מ־8. אם תוצאת הביניים היא 0, מסתיים תהליך החילוק והתקבלה תוצאה מדויקת.

		מונה
33		
— 8		
25	חיובי	0 + 1 = 1
— 8		
17	חיובי	1 + 1 = 2
— 8		
9	חיובי	2 + 1 = 3
— 8		
1	חיובי קטן מ־8	3 + 1 = 4

התוצאה הסופית של החילוק נמצאת, איפוא, בין 4 ל־5.

דיוק החילוק: כדי לקבל דיוק רב יותר, קובעים את הנקודה העשרונית מימין לתוצאה שהתקבלה, כופלים את תוצאת הביניים ב־10 וממשיכים עד לדיוק הרצוי.

נמשיך את הדוגמא הקודמת כדי לקבל תוצאה מדויקת יותר:

$1 \times 10 = 10$		מונה
-8		
2	חיובי קטן מ־8	$0 + 1 = 1$

כופלים ב־10 וממשיכים:

$2 \times 10 = 20$		מונה
-8		
12	חיובי	$0 + 1 = 1$
-8		
4	חיובי קטן מ־8	$1 + 1 = 2$

כופלים ב־10 וממשיכים:

$4 \times 10 = 40$		מונה
-8		
32	חיובי	$0 + 1 = 1$
-8		
24	חיובי	$1 + 1 = 2$
-8		
16	חיובי	$2 + 1 = 3$
-8		
8	חיובי	$3 + 1 = 4$
-8		
0	אפס	$4 + 1 = 5$

התוצאה 4.125

אם לא מורים למחשב להפסיק את הפעולה כאשר נתקבלה התוצאה הנכונה (נתגלתה שארית של אפס), הוא ימשיך בפעולה עד אשר יושג גבול הקיבולת של מונה התוצאה.

(לדוגמא: אם מונה התוצאה הוא בן 11 ספרות, שאחת מהן מיועדת לסימון ערכו השלילי או החיובי של המספר, תהיה התשובה לחילוק 33 ב־8 4.125000000 +
ברור ששיטה זו של חילוק באמצעות חיסור (למעשה באמצעות חיבור) גוזלת זמן רב. למשל, לביצוע "1000 לחלק ב־2" צריך לעשות 500 פעולות.

חשבון בינרי

כללי החיבור של חשבון בינרי:

1. 1 + 1 = 0 ונגררת של 1
2. 1 + 0 = 1
3. 0 + 0 = 0

חשוב לציין, שהשיטה הבינרית דורשת רק 3 כללים, בעוד שהשיטה העשרונית דורשת הרבה יותר.

נניח כי מבקשים לחבר 6 ו־3 בצופן הבינרי:

בינרי		עשרוני
0110	=	6
+ 0011		+ 3
1001		9

נבדוק את התוצאה:

$$1\times 2^3 + 0\times 2^2 + 0\times 2^1 + 1\times 2^0$$

$$8 \quad 0 \quad 0 \quad 1 \quad = 9$$

חיבור סידרתי וחיבור במקביל

חיבור סידרתי הוא הדרך המקובלת לחיבור מספרים: מחברים סיפרה סיפרה. כאשר המהירות חשובה משתמשים בחיבור במקביל אשר בו מחברים את כל הספרות בו־זמנית. ההבדל הוא עקרוני ביותר, ויש להבחין בכך.

חיבור סידרתי

	4	3	6
	+		
	2	1	7
צעד א׳			3(1)
		+1	
צעד ב׳		5	
צעד ג׳	6		
	6	5	3

חיבור במקבל

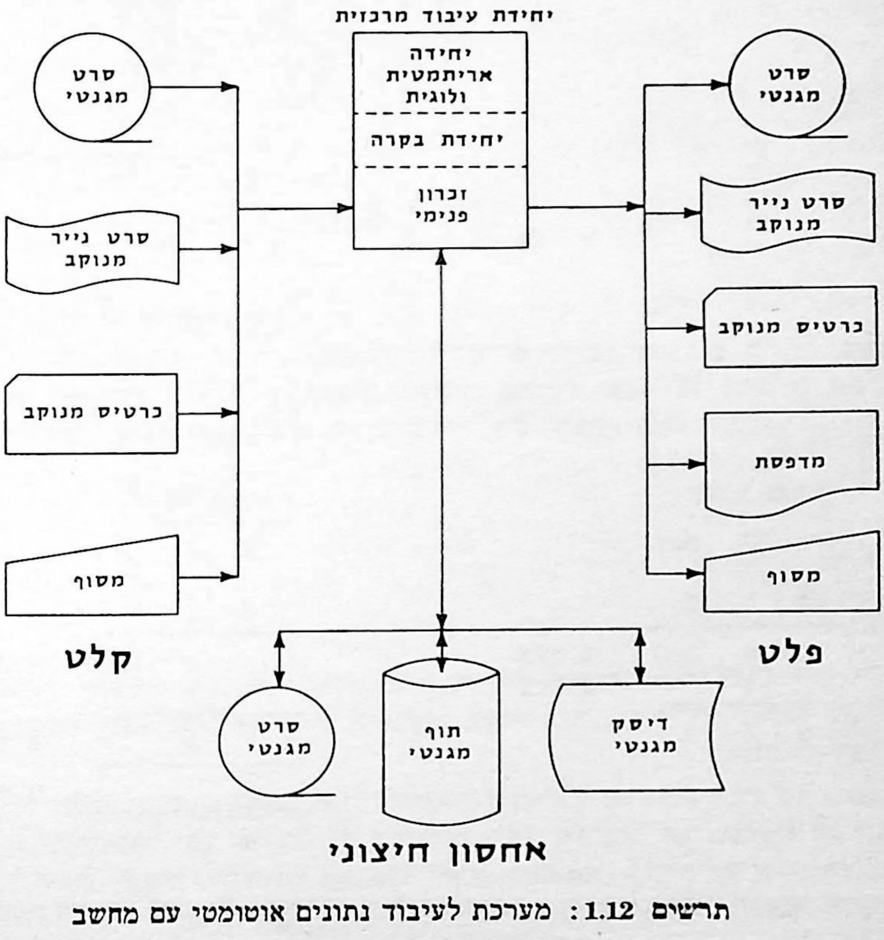
	4	3	6
	+		
	2	1	7
צעד א׳	6	4	3(1)
צעד ב׳		+1	
	6	5	3

בשיטת החיבור במקביל במספרים גדולים החסכון במספר הפעולות הוא רב יותר.

3.

מערכת המחשב

מערכת מחשב מורכבת מפריטי ציוד רבים ומגוונים, הנבחרים בהתאם לצרכי עיבוד הנתונים של הארגון בו מותקנת המערכת (תרשים 1.12). כפי שהוזכר בפרק 1 מחייב תהליך עיבוד הנתונים קיומן של מספר פונקציות: קלט, אחסון, עיבוד, בקרה ופלט (ראה תרשים 1.3). כל פונקציה כזו מתבצעת באמצעות פריט ציוד מסויים. הבקרה, הפיקוח וחלק מצומצם של האחסון מרוכזים ביחידת העיבוד המרכזית, וכל שאר הפונקציות מבוצעות על־ידי יחידות הקלט והפלט ויחידות האחסון החיצוני. מכלול פריטי הציוד במערכת נקרא חומרה (hardware).



תרשים 1.12: מערכת לעיבוד נתונים אוטומטי עם מחשב

החומרה איננה מסוגלת לפעול ולמלא את תפקידיה ללא תוכניות המפעילות את פריטי הציוד. החומרה יחד עם התוכניות הבסיסיות הם המרכיבים של מערכת המחשב ומהווים את המשאבים המאפשרים את תפעול המערכת.
יחידות הקלט והפלט מחוברות ליחידת העיבוד המרכזית. הן מופעלות על־ידי הוראות תוכנית ויכולות לקלוט ולהעביר נתונים ותוכניות אל ומן המחשב. יחידת קלט הנה יחידה מכנית־אלקטרונית המתרגמת את הנתונים המיועדים לעיבוד לאותות חשמליים ומעבירה אותם אל המחשב; יחידת פלט הנה יחידה מכנית־אלקטרונית המתרגמת את האותות החשמליים, המייצגים נתונים שעובדו על־ידי המחשב, לתוצאות קבועות, כגון דפים מודפסים, סימנים מגנטיים על סרט או על תקליט, כרטיסים מנוקבים. יחידות רבות יכולות לשמש הן כיחידות קלט והן כיחידות פלט בהתאם לצורך.
הבחנה חשובה בין יחידות הקלט/פלט הנה בהתאם לצורת הצגת הנתונים: חלק מיחידות הקלט/פלט מקבלות ומפיקות נתונים חזותיים, הניתנים להבנה על־ידי אדם — למשל דף מודפס, מסך המציג תרשים או דוחות. חלק אחר מיחידות הקלט/פלט מקבל ומפיק נתונים שאינם ניתנים לקריאה או הבנה באופן ישיר על־ידי אדם, למשל — סימנים מגנטיים על גבי סרט או תקליט מגנטי, ניקובים על גבי סרט נייר או כרטיס. יחידות הקלט/פלט, המקבלות או מפיקות נתונים שאינם ניתנים לקריאה על־ידי אדם הנן יחידות אחסון — המחשב כותב עליהן נתונים לצורך קריאת הנתונים מאוחר יותר על־ידי המחשב עצמו. יחידות אחסון אלה נקראות יחידות אחסון חיצוני.
מערכות המחשב הקיימות מאופיינות בהתאם לשיטת ביצוע העיבודים: רוב המערכות פועלות בשיטת העיבוד במכלול (batch processing); חלק מהמע־רכות — בעיקר המתקדמות והמתוחכמות יותר — מסוגלות לפעול בשיטת העיבוד הישיר (on-line processing) בנוסף לעיבוד במכלול.
בשיטת העיבוד במכלול — התוכנית וכל הנתונים הדרושים לתפעולה מרוכזים ביחד — בחבילת כרטיסים, על סרט מגנטי או על תקליט מגנטי ומעובדים במחשב ברצף אחד. בשיטת העיבוד הישיר, עיבוד הנתונים נעשה צעד אחר צעד, בכל צעד — נוספים נתונים או חלקי תוכנית המאפשרים את המשך העיבוד. התוצאה של כל צעד מוצגת מייד ומאפשרת את המשך הפעולה. בשיטת העיבוד הישיר ציוד הקלט והפלט קשור ישירות ליחידת העיבוד המרכזית, ומקבל ממנה נתונים המייצגים את המצב הנוכחי של התוכנית מייד בסיום כל צעד בעיבוד. ציוד הקלט והפלט בשיטת העיבוד הישיר הנו בדרך־כלל מסופים (terminals) הכוללים לוח מקשים המאפשר להזין את הנתונים אל המחשב, ומסך או נייר אשר עליהם מוצגות תוצאות העיבודים.

יחידת העיבוד המרכזית (Central Processing Unit — CPU)

היחידה הבסיסית במערכת המחשב הנה יחידת העיבוד המרכזית, המאפשרת פעולה של המערכת כולה. יחידה זו הנה למעשה המחשב המהווה את מרכז מערכת.

מהירות ביצוע הפעולות ביחידה זו גבוהה מאד: במחשבים הקיימים מתבצעות למעלה מ־1,000,000,000 פעולות בשניה!

יחידת העיבוד המרכזית מורכבת משלושה חלקים:

1. יחידת בקרה ופיקוח (control unit).
2. היחידה האריתמטית־לוגית (arithmetic-logic unit).
3. הזיכרון הפנימי (internal memory).

יחידת הבקרה מפקחת על יחידות הקלט וקובעת אילו נתונים לקרוא, מהיכן, ובאיזה מקום בזיכרון הפנימי לאחסן את הנתונים הללו. יחידה זו גם פוקדת על היחידה האריתמטית־לוגית אילו פעולות לבצע על הנתונים, מאיזה מקום בזיכרון לשלוף אותם, והיכן לאחסן את התוצאות. לבסוף, יחידת הבקרה גם מפקחת ופוקדת על אמצעי הפלט מה להפיק והיכן. כל זה נקבע בהתאם להוראות התוכנית המבוצעת באותה עת.

היחידה האריתמטית־לוגית מבצעת את פעולות החישוב וההשוואה של הנתונים (ראה פרק 1.2) בהתאם להנחיות התוכנית.

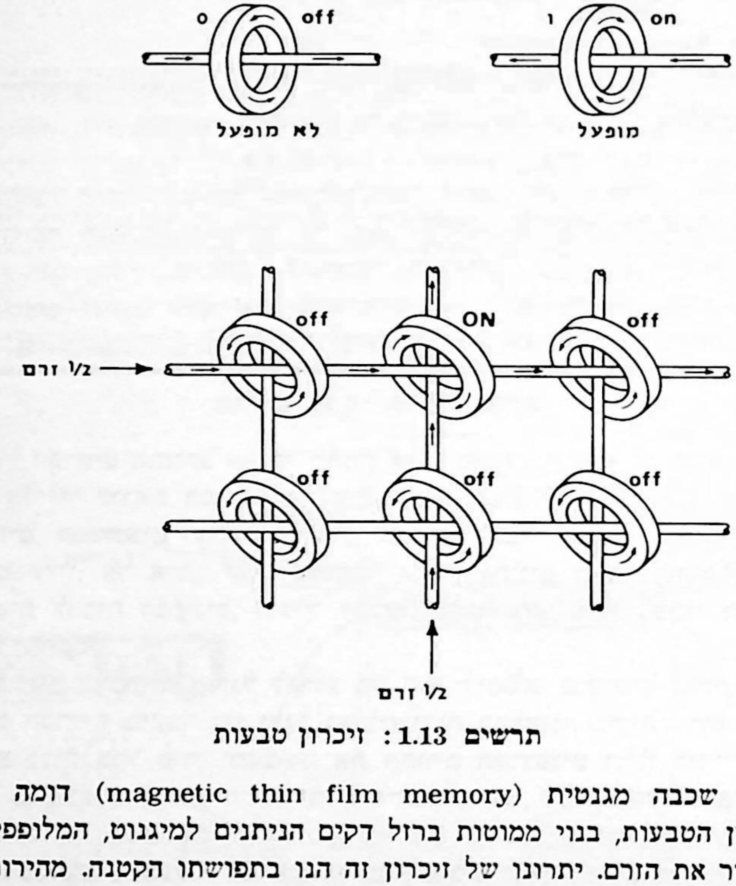
הזיכרון הפנימי מאחסן את הוראות הפעולה של התוכנית ואת הנתונים המיועדים לעיבוד על־ידי התוכנית. זיכרון זה מאופיין בזמן גישה קצר לנתונים ומהירות העברת נתונים גבוהה מאד. הזיכרון הפנימי בנוי מתאי זיכרון, המורכבים ממספר סיביות. מספר הסיביות בתא־זיכרון קובע את גודל מילת המחשב (כזכור, מילת המחשב הנה צירוף של תווים). קיבולת הזיכרון נקבעת על פי מספר התאים המוכלים בו; במחשבים הגדולים הקיימים קיבולת הזיכרון הנה כחמישה מיליון תאים. לכל תא כתובת המאפשרת לזהותו בעת הצורך. התוכנית יכולה לקרוא או לכתוב נתונים בתאי זיכרון אלה על־ידי הוראות מתאימות.

קיבולו של הזיכרון הפנימי מצומצם יחסית ואיננו מאפשר אחסון כל הוראות הפעולה והנתונים הדרושים לביצוע תוכנית, לכן מאוחסנים בו רק אותו חלק מהוראות הפעולה של התוכנית והלק מהנתונים הדרושים לעיבוד מידי. הוראות פעולה ונתונים שאינם דרושים לעיבוד מידי נשמרים ביחידות האחסון החיצוני, מועברים אל הזיכרון הפנימי כאשר הם נדרשים לעיבוד, ובגמר העיבוד הם מועברים חזרה אל יחידות האחסון החיצוני.

הזיכרון הפנימי בנוי מתאים הנותנים למיגנוט, המייצגים את הסיביות: המיגנוט יכול להיות באחד משני מצבים, המייצגים את המצבים 0 ו־1 של הסיבית. מהירות העברת הנתונים בזיכרון הפנימי הנה כמהירות זרימת החשמל (מהירות האור); זמן הגישה לתא זיכרון מסויים תלוי בגודלו הפיזי של הזיכרון, כלומר — בהתאם למרחקים בין תאי הזיכרון אותם צריך זרם החשמל לעבור כדי להגיע לתא הרצוי. טכניקות וחומרים שונים משמשים לבניית הזיכרון הפנימי במטרה לצמצם את גודלו.

במחשבים הקיימים הזיכרון הפנימי בנוי באחת משלוש השיטות:

— זיכרון טבעות (magnetic core memory) הבנוי ממיליוני טבעות ברזל הניתנות למיגנוט על־ידי זרם העובר דרכן. כל טבעת מייצגת סיבית אחת, שניתן לזהותה על פי מיקומה ברשת החוטים האנכיים והאפקיים. העברת זרם בעצמה מסויימת דרך שני החוטים המתאימים תמגנט את הטבעת הנמצאת בהצטלבות החוטים הללו. כיוון הזרם קובע את כיוון המיגנוט. טבעת ממוגנטת יכולה להשאר במצבה ללא הגבלת זמן. זיכרון זה מהיר ואמין, דבר שהביא לשימוש נרחב בו (תרשים 1.13).



תרשים 1.13: זיכרון טבעות

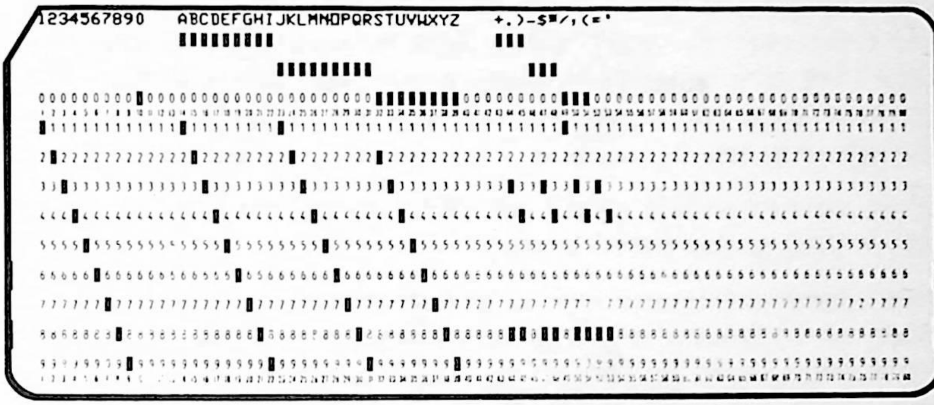
— זיכרון שכבה מגנטית (magnetic thin film memory) דומה בפעולתו לזיכרון הטבעות, בנוי ממוטות ברזל דקים הניתנים למיגנוט, המלופפים בסליל המעביר את הזרם. יתרונו של זיכרון זה הנו בתפושתו הקטנה. מהירותו גבוהה פי 2 עד 3 ממהירות זיכרון הטבעות.

— זיכרון מעגלים משולבים (monolitic integrated circuits) בנוי ממיליוני חלקיקי סיליקון היצוקים במתכת, פלסטיק או קרמיקה. תפושתו קטנה במיוחד — סנטימטר מרובע מכיל 512 סיביות. זיכרון זה אמין, מהיר, צורך פחות חשמל והוא זול יותר ביחס לסוגי הזיכרון האחרים.

יחידות קלט ופלט (Input and Output Devices)

1. קוראת ומנקבת כרטיסים (card reader and punch)

ציוד זה משתמש בכרטיסי קרטון מנוקבים. הכרטיס המקובל מחולק ל־80 טורים ו־12 שורות (תרשים 1.14). שתים־עשרה השורות בכל טור משמשות לציון התו המתאים לכל טור על־ידי ניקוב המקומות המתאימים בצופן מוסכם. ניתן לפענח את התו המנוקב בכל טור ולהדפיס אותו בראש הכרטיס. הפינה השמאלית־עליונה של הכרטיס חתוכה, כדי לוודא שהכרטיס מונח בצורה הנכונה בזמן הקריאה.



תרשים 1.14: כרטיס מנוקב

יתרונו של הכרטיס המנוקב — קל להכין אותו באמצעות מכונות ניקוב, הוא ניתן לקריאה על־ידי קוראת הכרטיסים, וכאשר הוא מפוענח — גם אדם יכול לקרוא את התווים המסומנים עליו. הכרטיס המנוקב יכול לשמש לאחסון הנתונים המנוקבים עליו, אך איננו יעיל כאמצעי אחסון נתונים מאחר שהוא תופס נפח גדול יחסית לכמות הנתונים, ובגלל הבלאי של הקרטון ממנו עשוי הכרטיס — המתעוות ומתקמט.

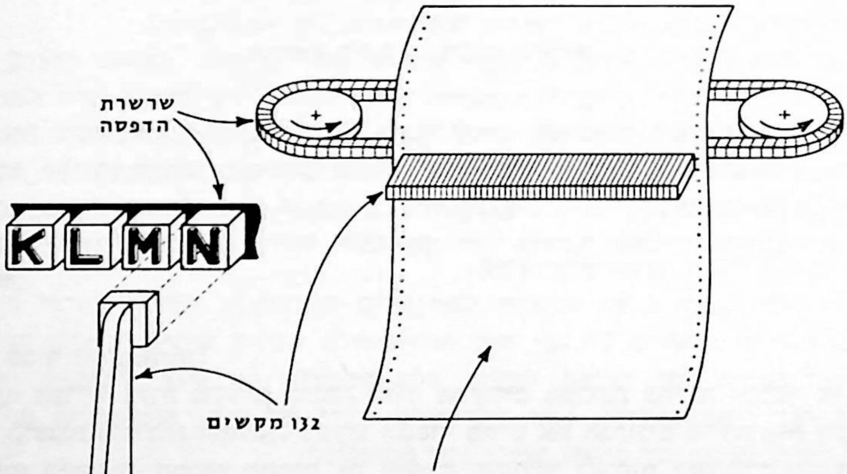
הציוד לקריאת כרטיסים מסוגל לקרוא בין 500 ל־2000 כרטיסים בדקה. בקוראת־הכרטיסים הכרטיס מועבר בין גלגל נע למברשת המבחינה בין השורות המנוקבות לבלתי מנוקבות בכל טור, ומעבירה את התווים המוצפנים הללו ליחידת העיבוד המרכזית באמצעות אותות חשמליים. מברשת קריאה שניה מוודאת את הקריאה שבוצעה על־ידי המברשת הראשונה (תרשים 1.15). קיימות קוראות־כרטיסים המבחינות בניקוב באמצעות תא פוטואלקטרי הרגיש לאור במקום מברשת.



תרשים 1.15: קוראת כרטיסים

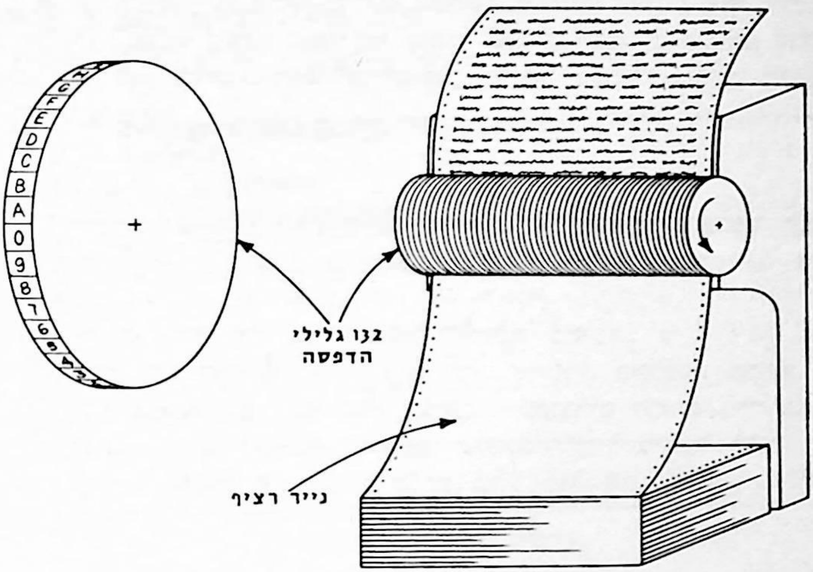
מנקבת הכרטיסים מסוגלת לנקב בין 60 ל־500 כרטיסים בדקה. מנקבת הכרטיסים הופכת את האותות החשמליים המתקבלים מן המחשב לניקובים על הכרטיס.
קיימות יחידות המיועדות לקריאה בלבד, לניקוב בלבד, ויחידות רב־תכליתיות המסוגלות לקרוא, לנקב ולפענח את הכרטיסים. בנוסף למנקבת הכרטיסים המחוברת למחשב, קיימות מכונות ניקוב שאינן מחוברות למחשב המיועדות להכנת כרטיסים מנוקבים לצורך קריאתם על־ידי המחשב. מכונות הניקוב מצויידות בלוח מקשים הדומה למכונת כתיבה, ועל־ידי הקשת התווים הרצויים נוצרים הניקובים על הכרטיסים. קצב הניקוב במכונות הניקוב על־ידי נקבנית מנוסה הנו כ־10 עד 15 כרטיסים בדקה.

2. מדפסת (line printer)
המדפסת הופכת את האותות החשמליים המייצגים את תוצאות העיבוד לאותיות וספרות מודפסות על גבי נייר. המדפסות מסוגלות להדפיס בין 300 ל־3000 שורות בדקה.
מדפסות הקשה פועלות על עקרון דומה למכונת כתיבה: נייר רציף מועבר במהירות בין שרשרת ההדפסה וסרט ספוג דיו. שרשרת ההדפסה מכילה מספר סדרות של התווים ומסתובבת במהירות. המקשים מסודרים לרוחב נייר ההדפסה, בהגיע התו הרצוי למקום המיועד להדפסתו על גבי הנייר מופעל מגנט חשמלי המכה על המקש ומצמיד את הנייר לסרט הדיו מול התו אותו יש להדפיס (תרשים 1.16).



תרשים 1.16: מדפסת שרשרת

מדפסת הקשה אחרת פועלת בצורה דומה, אך במקום שרשרת קיימים בה גלילים ובהם תווי ההדפסה. הגלילים סובבים במהירות וכאשר התו הרצוי נמצא מול המקום המיועד להדפסה — מכה המקש על הנייר ומצמיד אותו אל סרט הדיו מול התו אותו יש להדפיס (תרשים 1.17).



תרשים 1.17 : מדפסת גלילים

מדפסות מסוג אחר, המעבירות את התו לנייר באמצעים חשמליים או טרמיים במקום על־ידי הקשה, מהירות בהרבה ממדפסות ההקשה — מדפיסות כ־5000 שורות בדקה. חסרונן בכך שאינן יכולות להדפיס נייר במספר העתקים, איכות ההדפסה ירודה, והנייר הדרוש לפעולתן יקר בהרבה מהנייר הדרוש למדפסת הקשה.

3. מסוף (terminal)

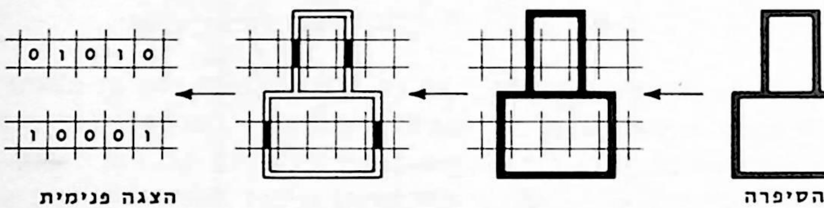
המסוף מצוייד בלוח מקשים הדומה ללוח המקשים במכונת כתיבה ובמסך או נייר עליו מוצגות תשובות המחשב. מפעיל המסוף מקיש את הנתונים או הוראות הפעולה על לוח המקשים, המסוף מתרגם את התווים שהוקשו לאותות חשמליים המועברים למחשב, וכאשר מתקבלת תגובת המחשב — מתרגם המסוף את האותות החשמליים המתקבלים לתווים המוצגים על גבי המסך או המודפסים על גבי הנייר.
באמצעות מסוף כזה מפקח מפעיל מערכת המחשב על פעילות המערכת; מסוף זה נקרא console.

4. יחידות לקריאה אופטית או מגנטית

קיימות מספר שיטות בהן הציוד יכול לקרוא נתונים ישירות ממסמכים, ובכך לחסוך את שלב הכנת הנתונים למחשב.

קריאת סימנים (mark reading): בשיטה זו הטפסים המיועדים לקריאה מחולקים למשבצות, וממלא הטופס מסמן בעפרון או עט רגיל את המשבצות המתאימות לתווים הרצויים בכל שורה ועמודה. הטופס המסומן עובר דרך היחידה לקריאת הסימנים, הסורקת את הטופס ומתרגמת את המשבצות הריקות והמסומנות לאותות חשמליים המועברים אל המחשב.

קריאת תווים אופטית (optical character recognition): בשיטה זו הנתונים נרשמים על הטפסים על־ידי שימוש באותיות וספרות בעלות מבנה מוסכם וקבוע. הטופס עובר דרך היחידה לקריאת תווים, הקוראת כל תו בצורה אופטית, מחלקת את "התמונה" שהתקבלה למשבצות או לנקודות, ומתרגמת את המשבצות הבהירות והכהות לאותות חשמליים (תרשים 1.18).



תרשים 1.18: קריאה אופטית של הסיפרה 8

קריאת תווים מגנטית (magnetic ink character recognition):
בשיטה זו נרשמים התווים על גבי הטפסים בדיו מגנטית. הטופס מועבר דרך היחידה לקריאת תווים מגנטיים, המקבלת זרם חשמלי היחסי לכמות הדיו המגנטית בכל תו. יתרונה של השיטה הוא שניתן לזהות את התווים שנכתבו בדיו מגנטית גם אם הודפסו עליהם תווים אחרים (למשל — חותמת בנק המוטבעת על המחאה שנכתבה בדיו מגנטית). שיטה זו מקובלת כיום ברוב הבנקים לסימון המחאות.

5. יחידות להכנת סרטי מיקרופילם

יחידות אלה מעבירות את הנתונים המתקבלים מן המחשב ישירות לסרטי מיקרו־פילם. התווים מסומנים על גבי סרט המיקרופילם בצורת אותות וספרות הניתנות לקריאה בעזרת מסך הגדלה. יחידות אלה מעבירות בין 7000 ל־30,000 שורות בדקה לסרטי המיקרופילם. סרט מיקרופילם בגודל של כ־10 סנטימטרים מרובעים יכול להכיל תוכנם של כ־300 עמודי נייר מודפסים!

6. תווין (graph plotter)

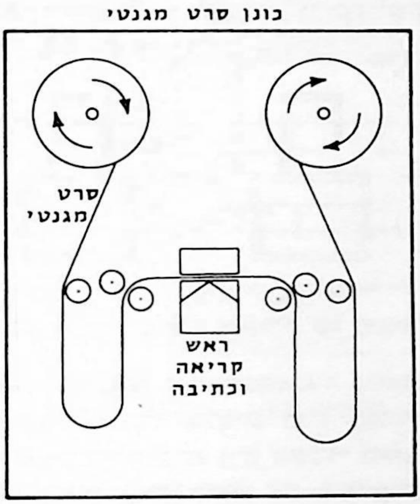
התווין מתרגם את האותות החשמליים המתקבלים מן המחשב לשרטוט על גבי נייר רציף. התווין שימושי בעיקר בהכנת גרפים ותרשימים שהם תוצאות של עיבודים במחשב.

יחידות אחסון חיצוני (Auxiliary Storage)

1. סרט מגנטי (magnetic tape)

הסרט המגנטי עשוי מחומר פלסטי מצופה בחומר הניתן למיגנוט. לצורך קריאה או כתיבה על הסרט המגנטי מורכב גליל הסרט על כונן, המעביר את הסרט מול ראש קריאה־כתיבה. הכתיבה על הסרט נעשית על־ידי מיגנוט נקודות בהתאם לתווים אותם יש לכתוב, וקריאת הסרט נעשית על־ידי קריאת נקודות אלה והפיכתן לאותות חשמליים (תרשים 1.19).

כונני הסרט המגנטי מאופיינים על־ידי מהירות הקריאה והכתיבה על גבי הסרט וצפיפות הנתונים הנכתבים על הסרט. מהירות הקריאה והכתיבה מהסרט המגנטי הנה כ־350,000 תווים בשניה.



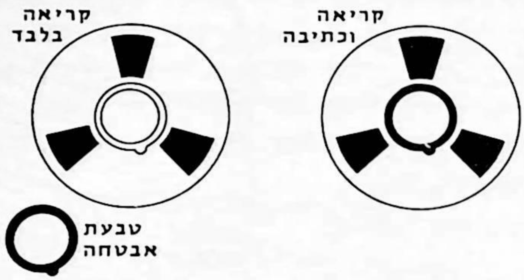
תרשים 1.19: כונן סרט מגנטי

הכתיבה על גבי הסרט נעשית בצופן בינרי, כאשר 1 מיוצג על־ידי נקודה ממוגנטת ו־0 מיוצג על־ידי נקודה שאיננה ממוגנטת. התווים נכתבים על הסרט בצופן בינרי מיוחד המבוסס על 7 או 9 סיביות — כמספר הערוצים בסרט (תרשים 1.20). צפיפות רישום הנתונים על גבי הסרט הנה כ־1600 תווים באינטש.



תרשים 1.20: סרט מגנטי

כתיבה על גבי הסרט מוחקת את הנתונים הקודמים הכתובים על הסרט באותו מקום. כדי למנוע מחיקת נתונים בטעות, קיימת טבעת אבטחה אותה יש להצמיד לגליל הסרט כדי לאפשר כתיבה עליו. בהעדרה של טבעת האבטחה ניתן רק לקרוא מהסרט. כדי לאפשר כתיבה על הסרט הייב מפעיל המערכת להרכיב את הטבעת בגליל הסרט בזמן הרכבת הגליל על הכונן (תרשים 1.21).



תרשים 1.21: גליל סרט מגנטי וטבעת אבטחה

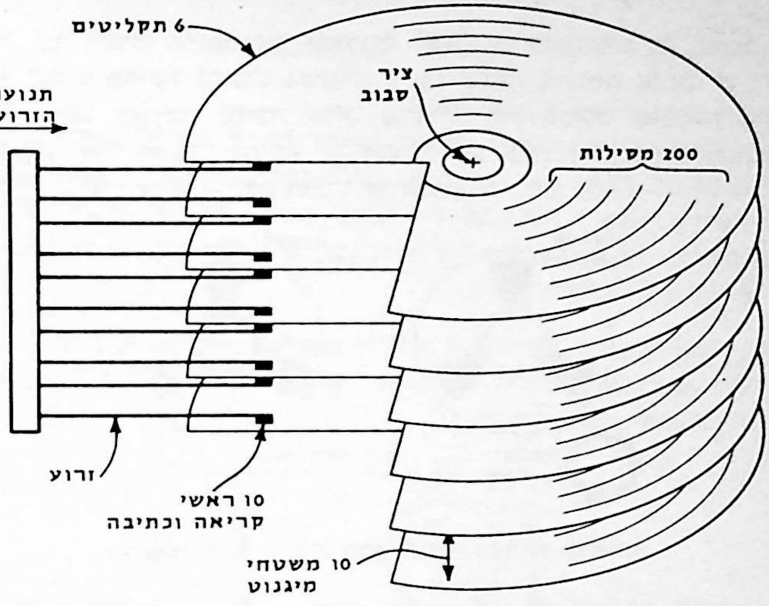
יתרונותיו של הסרט המגנטי הם: מהירות גבוהה בקריאה וכתיבת נתונים, קיבולת גדולה בנפח מצומצם, מחיר זול, ואפשרות לכתוב ולקרוא ממנו למעלה מ־50,000 פעם. חסרונו בכך שהוא דורש רישום הנתונים בסדר עוקב, דבר המונע אפשרות להוספת נתונים חדשים לתוך סידרת נתונים קיימת. חסרון אחר הנו שהנתונים הרשומים על גבי הסרט אינם ניתנים לקריאה על־ידי אדם.

2. תקליט מגנטי (magnetic disk)

התקליט המגנטי בנוי משישה או עשרה תקליטים המחוברים לציר אחד. התקליטים עשויים מחומר מתכתי הניתן למיגנוט. לכל תקליט שני משטחי מיגנוט, וכל משטח מחולק ל־200 מסילות — שהן מעין טבעות סביב מרכז התקליט — עליהן נכתבים הנתונים. לכל משטח מיגנוט מתאימה זרוע אשר בקצה מורכב ראש הקריאה־כתיבה; הזרוע מסוגלת לנוע לכיוון מרכז התקליט או להיקף התקליט, ולהביא את ראש הקריאה־כתיבה מול המסילה הרצויה (תרשים 1.22). קבוצת המסילות הנמצאות אחת מעל השניה במשטחי הקריאה נקראת גליל.

התקליט המגנטי מורכב בכונן התקליטים ומסתובב במהירות גבוהה — כ־1800 סיבובים בדקה. הנתונים כתובים לאורך המסילות ונקראים או נכתבים על־ידי ראש הקריאה־כתיבה בזמן סיבוב התקליט. הנתונים נשמרים על גבי התקליט בצופן בינרי בצורת נקודות ממוגנטות ולא־ממוגנטות.

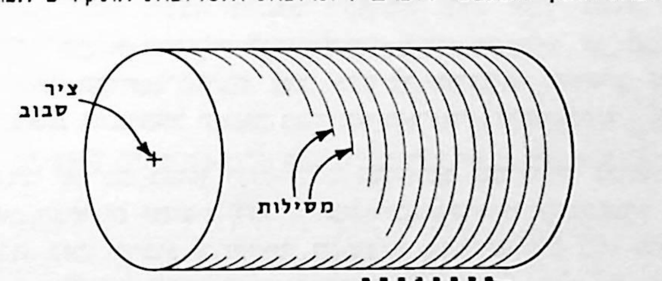
יתרונותיו של התקליט המגנטי — מהירות גבוהה בקריאת וכתיבת נתונים, זמן גישה לנתונים קצר, קיבולת גדולה, ואיננו דורש רישום הנתונים בסדר עוקב (בניגוד לסרט המגנטי). חסרונו של התקליט המגנטי במחירו הגבוה.



תרשים 1.22:

3. תוף מגנטי (magnetic drum)

התוף המגנטי בנוי מגליל מתכתי המצופה בחומר הניתן למיגנוט. היקפו החיצוני של הגליל מחולק למסילות עליהן נרשמים הנתונים בצופן בינרי על־ידי מיגנוט או אי־מיגנוט נקודות לאורך המסילות (תרשים 1.23). התוף המגנטי מורכב בכונן, בו הוא מסתובב במהירות גבוהה; מול כל מסילה נמצא ראש הקריאה־כתיבה המסוגל לקרוא או לכתוב את הסימנים המגנטיים שעל גבי התוף תוך כדי סיבוב התוף. בניגוד לתקליט המגנטי, ראשי הקריאה־כתיבה של התוף המגנטי הינם קבועים — לכל מסילה ראש קריאה־כתיבה — דבר המקצר את זמן הגישה לנתונים. יתרונות וחסרונות התוף המגנטי דומים ליתרונות וחסרונות התקליט המגנטי.

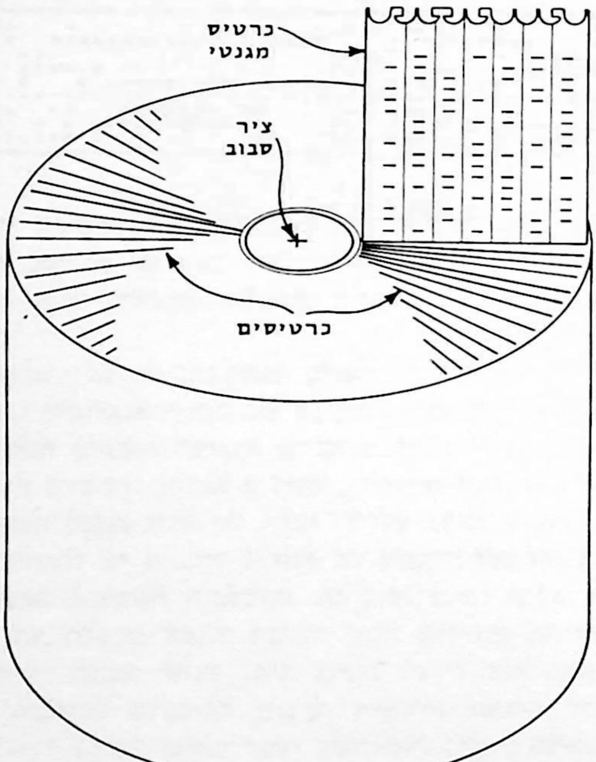


תרשים 1.23:

.4 Card Random Access Memory — CRAM

התקן זה בנוי ממיכל עגול, המחולק לתאים המכילים כרטיסים פלסטיים שעליהם דבוקים קטעי סרט מגנטי. קטעי הסרט המגנטי ממוגנטים בהתאם לנתונים אותם יש לשמור — כפי שתואר לסרט המגנטי.

ה־CRAM מורכב בכונן, בו המיכל יכול להסתובב ולהביא את הכרטיס הרצוי אל מול זרוע־שליפה, המוציאה את הכרטיס מן המיכל, מעבירה אותו לראש הקריאה־כתיבה, אשר קורא או כותב עליו את הנתונים הדרושים, ולאחר מכן מחזירה זרוע השליפה את הכרטיס חזרה אל התא אליו הוא שייך (תרשים 1.24) הזמן הדרוש לסיבוב המיכל ולשליפת הכרטיס והחזרתו למקומו ארוך יחסית, וזה חסרונו העיקרי של ה־CRAM. התקן זה מתאים במיוחד לאחסון כמות גדולה של נתונים אשר מטפלים בהם מעט ולעיתים רחוקות.



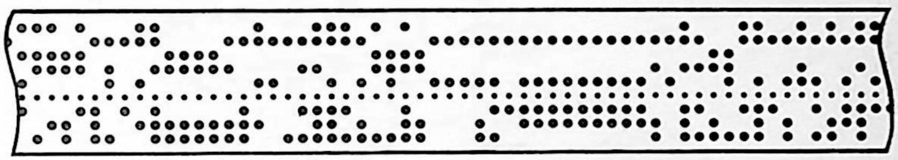
תרשים 1.24 : CRAM

.5 זיכרון לייזר (laser mass memory)

לאחרונה פותח זיכרון בעל קיבולת גדולה המבוסס על רישום נתונים באמצעות קרן־לייזר ומערכות אופטיות. התקן זה מסוגל להכיל כמות גדולה ביותר של נתונים בנפח מצומצם, כך שעלות האחסון לנתון נמוכה יחסית. יתרונות אחרים

של זיכרון לייזר הנם אמינותו הגבוהה ברישום ואחסון הנתונים, והעדר חלקים נעים.

6. סרט נייר מנוקב (punched paper tape)
בשיטה זו מנוקבים הנתונים על גבי סרט נייר, בדומה לסימונים מגנטיים על סרט מגנטי. כל נקודה לאורך סרט הנייר יכולה להיות מנוקבת או לא מנוקבת, מצבים המבטאים את הסיביות 0 ו־1. כל תו מבוטא בצופן בינרי של 7 סיביות (תרשים 1.25). יתרונותיה של שיטה זו במחירה הנמוך; חסרונותיה — זמן קריאה וכתיבה ארוך מאד, השימוש בסרט הנייר הנו חד־פעמי, ולא ניתן לשנות או להוסיף נתונים לנתונים הנמצאים על סרט הנייר.



תרשים 1.25: סרט נייר מנוקב

.4
תוכנה (Software)

התוכנה (software) הנה מכלול התוכניות הקיימות במערכת המחשב, ואילו החומרה (hardware) הנה מכלול הציוד הקיים במערכת המחשב. התוכנה גבנית עבור מערכת המחשב במסגרת יתרונות ומגבלות החומרה, וקיים קשר הדוק בין התוכנה לבין החומרה במערכת.
סביר להניח כי לאדם שאינו קשור לעולם המחשבים, המונח "תוכנה" הוא משהו ערטילאי. התוכנה היא "חשיבת" המחשב; אם נערוך אנאלוגיה לגוף האדם הרי התוכנה בעיקרה היא מה שמהווה במכניזם האנושי המנגנון המפקח על קבלת אותות מבחוץ, קבלת פקודות ופיענוחן, ותרגום אותות חיצוניים לאותות פנימיים שביכולתם לגרום לביצוע פעולה על־ידי הגוף. זהו אותו מנגנון המפקח על תיאום פעולות סימולטניות ועל סדר הביצוע של פעולות עוקבות. יש להבחין בין "חשיבה" לבין המוח, שהוא מקום משכן החשיבה. ההשלכות הנובעות מאנאלוגיה זו — חשיבה לקויה מפחיתה מיעילות הגוף ותוכנה פגומה מבטלת את ערך המחשב ויתרונו. כשם שאדם לומד ורוכש ניסיון, ניתן לשפר תוכנה ולייעל אותה; זהו תהליך מתמשך של הוצאת גירסאות חדשות לתוכניות (לתוכנה כמושג כולל תפקידים נוספים שאינם ניתנים לאנאלוגיה ואשר יפורטו להלן.)
מערכת תוכנה בסיסית למחשב מכילה כמה סוגי תוכניות:

1. מערכת ההפעלה — תוכנית הפיקוח הראשית (Operating System).
2. תוכניות תרגום לשפות תיכנות (Assemblers, Compilers).
3. תוכניות שירות (Utility Programs).

בנוסף למערכת התוכנה הבסיסית שתתואר להלן, תוכניות הנכתבות על־ידי המשתמש הן המבצעות את היישומים של הארגון, תוך כדי שימוש במערכת התוכנה הבסיסית ותחת פיקוחה (ראה להלן חלק 11).

מערכת ההפעלה היא התוכנית המופקדת על ארגון העבודה במחשב. ההבחנה העיקרית בין סוגי מערכות ההפעלה הנה בהתאם למספר המשתמשים שהיא יכולה לשרת (כלומר — בהתאם למספר התוכניות השונות בהן יכולה מערכת ההפעלה לטפל באותו זמן).

מערכת ההפעלה הפשוטה משרתת תוכנית יחידה, כאשר אין כל מתחרים אחרים של שירותיו הפנימיים של המחשב. מערכת הפעלה כזו נקראת חד־תוכניתית. מערכת הפעלה משוכללת יותר מסוגלת לשרת בו־זמנית מספר גדול של תוכניות, ונקראת רב־תוכניתית (multiprogrammed). מערכת ההפעלה מטפלת בהקצאת משאבי מערכת המחשב בין התוכניות, במטרה להשיג ניצול מקסימלי של המש־אבים. מידת הסיבוך במערכת הפעלה רב־תוכניתית גדולה לאין ערוך ממערכת ההפעלה החד־תוכניתית, מאחר שקיימת תחרות בין התוכניות הרבות על משאבי מערכת המחשב ובמיוחד על הזמן בו יחידת העיבוד המרכזית עומדת לרשות כל תוכנית.

הבחנה בין מערכות הפעלה רב־תוכניתיות נעשית בהתאם לשיטת הקצאת יחידת העיבוד המרכזית בין המשתמשים. שתי השיטות העיקריות הן זמן אמיתי (real-time) ושיתוף זמנים (time-sharing).

בשיטת הזמן האמיתי עובר השימוש ביחידת העיבוד המרכזית מתוכנית אחת לשניה רק לאחר שהתוכנית הראשונה משלימה את עבודתה ביחידת העיבוד; בשיטת שיתוף הזמנים — עובר השימוש ביחידת העיבוד המרכזית בכל פרק זמן קבוע.

דוגמא פשוטה לעבודה במערכת הפעלה רב־תוכניתית מובאת להלן:

נניח שבמערכת המחשב שלושה משאבים — יחידת עיבוד מרכזית, מדפסת וזיכרון חיצוני (תקליט מגנטי), ונניח שארבע תוכניות מתחרות על משאבים אלה בו־זמנית. תוכנית א׳ מעסיקה את יחידת העיבוד המרכזית, תוכנית ב׳ מעסיקה את הזיכרון החיצוני, תוכנית ג׳ מעסיקה את המדפסת, ותוכנית ד׳ מבקשת להעסיק את יחידת העיבוד המרכזית (המועסקת על־ידי תוכנית א׳). במצב המתואר, כל משאבי המערכת מנוצלים, ושלוש מבין ארבע התוכניות מקבלות — בו־זמנית — את השירות הנדרש להן. במערכת הפועלת בשיטת הזמן האמיתי, תוכנית ד׳ תמתין בתור ליחידת העיבוד המרכזית עד שתוכנית א׳ תסיים להעסיק אותה; בשיטת שיתוף הזמנים, תחליף מערכת ההפעלה את הקצאת יחידת העיבוד המרכזית לסירוגין מתוכנית א׳ לתוכנית ד׳ בכל פרק זמן קבוע, עד שאחת מהתוכניות או שתיהן יסיימו את עבודתן ביחידה זו. מהדוגמא ניתן לראות שבשיטת שיתוף הזמנים מקבלת כל תוכנית שירותים כאילו היא משתמשת יחידה במשאבים — תוכנית ד׳ איננה ממתינה לסיום העבודה ע״י תוכנית א׳, אלא שתיהן נהנות משירותי יחידת העיבוד המרכזית.

דוגמא פשוטה זו אינה אלא טיפה בים המורכבות; די להגדיל את כמות המשאבים ל־20, ומספר התוכניות ל־30, ולזכור שדרישות התוכניות למשאבים משתנות מרגע לרגע, כדי לקבל מושג על מידת מורכבותה של מערכת ההפעלה.

בנוסף להקצאת המשאבים בין התוכניות, מבצעת מערכת ההפעלה מספר תפקידים אחרים, שהעיקריים הנם:

1. טיפול בהתערבויות: התערבות הנה בקשה של תוכנית למשאב מסויים או סיום השימוש במשאב על ידי התוכנית. כאשר התוכנית מבקשת להשתמש במשאב כלשהו, מטפלת מערכת ההפעלה במיקומה של התוכנית בתור המבקשים של אותו משאב. כאשר התוכנית מסיימת את השימוש במשאב — מטפלת מערכת ההפעלה בהקצאת המשאב שהתפנה לתוכנית הבאה בתור לאותו משאב.
התערבויות מסוג אחר הנן תקלות למיניהן בתוכניות או במשאבים, אשר מערכת ההפעלה יודעת לטפל בהן בהתאם לכללי ההתגברות שלה.
2. קליטת עבודות: מערכת ההפעלה קולטת את נתוני הפיקוח של העבודות, הופכת אותם לטבלאות סטנדרטיות, מכינה אותם לעבודה וטוענת אותם בזיכרון, כך שהם מוכנים לשמש כקלט למנגנון הפיקוח על ביצוע התוכניות.
3. פיקוח על ביצוע התוכניות: מערכת ההפעלה מפקחת על הקצאת המשאבים הדרושים לתוכנית בתיאום עם תוכניות אחרות הפועלות באותו זמן ומפקחת על שלמות הנתונים והביצוע ועל מניעת ערבוב הנתונים והדרישות של התוכניות השונות.

מערכות ההפעלה מהוות נושא רחב ומורכב, ואין זה המקום להכנס לכל פרטיהן. מאמצים רבים מושקעים בשכלול מערכות ההפעלה; הן מצריכות כוח אדם מעולה לתחזוקן ועדכונן, ואנשי תיכנות בשטח זה (system programmers) הנם המובחרים שבתוכניתנים.

תוכניות תרגום (מהדרים) מתרגמות משפת תיכנות עילית לשפת פקודות מחשב. יצרני התוכנה מספקים שפות תיכנות "עיליות", שהתיכנות בהן קל ופשוט יחסית. פקודות בשפות אלו דומות להוראות ביצוע "אנושיות" והן מבצעות פעולות מורכבות, שתכנותן בשפה הבסיסית של המחשב, הבנויה מצופן בינרי, היה גוזל זמן רב. לכל מחשב צמודה שפה בסיסית (הנקראת שפת מכונה). תוכנית בשפת המכונה הנה סידרת ההוראות האלמנטריות לביצוע פעולות על־ידי המחשב.

תוכניות שירות הנן אוסף של תוכניות שאין להן זיקה למשתמש מסוים, אלא נושאות אופי של פונקציות כלליות, כמו מיון, העברה של נתונים מכרטיסים או כל הנושא של תחזוקת קבצים (קובץ הנו אוסף מסודר של נתונים). בכל מערכת ממוחשבת קיימת ספרייה של תוכניות כאלה.

מחיר התוכנה למחשב גבוה ביחס למחיר החומרה; קיימת מגמת גידול במחיר התוכנה, וזאת משום שמחיר כוח אדם מיומן בשטח זה גבוה. נראה שמחיר התוכנה יהיה בעתיד בעל משקל עיקרי לעומת מחיר הציוד. עובדה זו מאלצת את המנהלים הפונים לעיבוד נתונים ממוכן במפעליהם להקדיש תשומת לב מרובה לכוח האדם המקצועי העומד לרשותם.

.5
תקשורת (Communication)

תאוריית התקשורת (Communication Theory)

שיטות שונות של תקשורת נחקרו ונלמדו במשך שנים רבות, אולם רק לאחרונה הוחל בחקר תהליך התקשורת עצמו.
לימוד וחקר תהליך התקשורת הנו חלק מתורה חדשה, הנקראת תאוריית התקשורת.
את המחקר היסודי הראשון בשטח זה עשה קלוד א. שנון (C. E. Shannon), שהניח למעשה את מרבית היסודות למה שידוע כיום כתאוריית התקשורת והביא תחת מכנה משותף את ההיבטים השונים שלה.
המרכיבים הבסיסיים של כל תהליך תקשורת הם מקור ההודעה, אמצעי העברת ההודעה והמקבל. התהליך הפשוט ביותר הוא כאשר אדם אחד מדבר אל אדם אחר. האדם המדבר הוא למעשה משדר ההודעה או המקור, האוויר משמש כאמצעי להעברת ההודעה והאדם השומע משמש כמקבל ההודעה. צורה כללית זאת נכונה לכל שיטות התקשורת הקיימות, כגון מערכת הטלפון, הרדיו, והטלוויזיה. כל השיטות משתמשות בשלושת המרכיבים הבסיסיים אשר צוינו לעיל.
כדי ליצור קשר, חייב מידע כלשהו להיות מועבר. "מידע", מוגדר בתורת התקשורת ככל סימן מוגדר. זה יכול להיות סידרת אותיות, קבוצה של טונים מוסיקליים, קבוצת תמונות וכד'. מידע זה מהווה הודעה (message).
כדי שתקשורת תהיה יעילה, היא חייבת להיות מובנת. במערכת התקשורת הפשוטה, ההודעה נשלחת בכיוון אחד בלי כל ידיעה באם המקבל מבין אותה או אינו מבינה. לדוגמא, באם אדם המקבל הודעה באמצעות טלפון אינו מבין חלק מההודעה, הוא יבקש לחזור על החלק הבלתי־מובן. אם חזרה זו אינה ניתנת לביצוע, הרי שההודעה שתתקבל לא תהיה מלאה, או לא תובן בצורה הנכונה.
האמצעי המשמש להעברת המידע הנו מוגבל. לדוגמא, ספר זה מוגבל במספר

העמודים וכמו כן בגודל. אמצעי העברת המידע בתחום המגבלות נקרא **ערוץ** (channel); בדיבור בין בני אדם ערוץ התקשורת הוא האוויר, במערכת טלפונים ערוץ התקשורת הוא מוליך חשמל, בעוד שבשידור רדיו או טלוויזיה הערוץ הוא תחום תדירות מסוים.

רעש (noise) הוא גורם חשוב בתהליך התקשורת. רעש מוגדר ככל גורם אשר מפריע להעברת המידע בצורה נקייה ונכונה. אנו נתקלים ברעשים כאלה יום יום, אם במערכת הטלפונים, עת נשמעות שיחות זרות או סתם רעשים שעה שאדם מדבר עם זולתו, "שלג" על מסך הטלוויזיה הנוצר עקב רעשים, אשר מפריעים לקליטה תקינה, וכדומה.

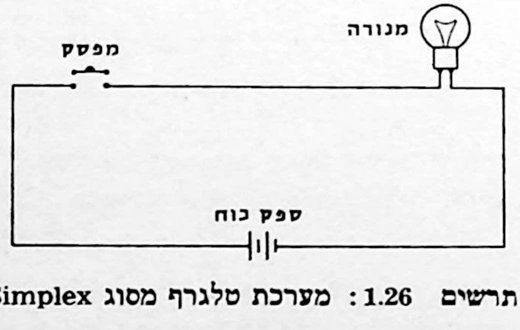
מערכות תקשורת

המאפיין העיקרי במערכות תקשורת הוא הזמן הדרוש להעברת המידע. **מערכות זמן אמיתי** הן מערכות תקשורת אשר בהן מתקבלת ההודעה בזמן האפשרי הקצר ביותר; אדם המקשיב לאדם שני בשיחת טלפון מקבל את ההודעה בזמן אמיתי. כלומר הוא מקבל אותה מיד ללא השהיות. כאשר נשלחת ההודעה על גבי מכתב, אין היא מתקבלת מיד. כלומר, מערכות כאלו אינן פועלות בזמן אמיתי.

רוב מערכות התקשורת של זמן אמיתי הן מערכות חשמליות או אלקטרוניות. טלפונים, רדיו וטלוויזיה הם דוגמאות למערכות זמן אמיתי.

חלוקה נוספת של מערכות תקשורת נעשית בהתאם לאפשרויות העברת המידע בין המקור והיעד: מערכת תקשורת אשר אינה מאפשרת קשר דו־כיווני בין המקור והיעד נקראת "simplex". מערכת המאפשרת קשר דו־כיווני נקראת "**duplex**". כדי להדגים מערכות תקשורת כאלה נשתמש בטלגרף, שהוא מערכת התקשורת החשמלית הראשונה שהומצאה הפועלת בזמן אמיתי.

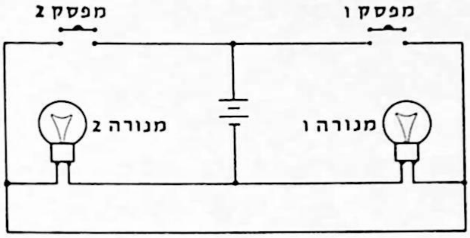
מערכת טלגרף הבנויה כמערכת simplex מוצגת בתרשים 1.26. במערכת זו מקור ההודעה הוא המפסק והיעד הוא המנורה. סגירת המפסק גורמת להדלקת המנורה. ליעד אין כל אפשרות לשדר הודעות אל המקור, ולכן איננו יכול לאמת או לברר דבר בקשר להודעה.



תרשים 1.26 : מערכת טלגרף מסוג Simplex

בצורה זאת ניתן לשלוח הודעות לפי סימנים מוסכמים (כמו שיטת המורס). אולם כאשר אין המנורה נמצאת בטווח עין של מפעיל המפסק לא יכול המפעיל לדעת אם המנורה פועלת, או במלים אחרות אם נתקבלה ההודעה שנשלחה על ידו.

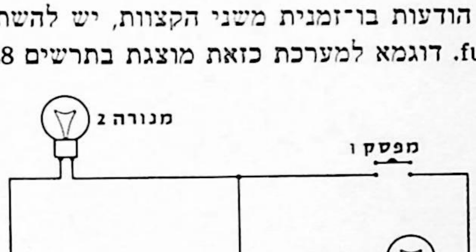
תרשים 1.27 מציג מערכת טלגרף פשוטה אחרת אשר בה ניתן לשלוח ולקבל הודעות בשני קצוות המערכת. מערכת זאת נקראת half duplex.



1.27 : מערכת טלגרף מסוג Half Duplex

במערכת זאת סגירת כל אחד מהמפסקים תגרום להדלקת שתי המנורות בעת ובעונה אחת. לכן ניתן להפעיל בזמן מסוים מפסק אחד בלבד, זאת כדי למנוע בלבול בין ההודעות הנשלחות משני המקורות (המפסקים).

אם יש צורך לשלוח הודעות בו־זמנית משני הקצוות, יש להשתמש במערכת טלגרף הנקראת full duplex. דוגמא למערכת כזאת מוצגת בתרשים 1.28.



תרשים 1.28 : מערכת טלגרף מסוג Full Duplex

במערכת זאת ניתן להפעיל את המפסקים באותו זמן כיוון שכל אחד מפעיל מנורה אחרת. תנועת ההודעות במערכת הזאת היא דו־סטרית וקיימים בה למעשה שני זוגות של מקורות ויעדים.

בדרך כלל ניתן להשתמש במערכות התקשורת הקיימות להעברת נתונים. המערכת המקובלת ביותר היא מערכת הטלפונים הרגילה. ניתן כמובן להשתמש גם במערכות שידור להעברת נתונים.

אמצעי תקשורת

אמצעי קשר קיימים

קווי טלפון פרטיים:
קווי טלפון פרטיים משתמשים ברשת הטלפונים הרגילה בה נקבע קו אחד כמיוחד למשתמש, בו ניתן להעביר נתונים ממקור ליעד קבועים, אך לא ניתן להתקשר באמצעות קו זה לקווים אחרים המחוברים לרשת הטלפונים. החיוב עבור קווים אלה הוא לפי תעריף מיוחד ונקבע בדרך כלל לפי ק״מ לחודש.

קווי טלפרינטר פרטיים:
קווי טלפרינטר פרטיים נמסרים בחכירה על־ידי חברת הטלפונים לשימוש המנויים. לרשת קווי הטלפרינטר ניתן להבר מסופים להעברת וקבלת נתונים ממקורות ליעדים שונים המחוברים לרשת. בחירת היעד נעשית על־ידי הקשת סימנו או מספרו, בדומה לחיוג ברשת הטלפונים.

אמצעי קשר מתמחים

האמצעים הקיימים כיום לתקשורת יועדו ותוכננו בעיקר עבור ערוצי דיבור. למרות הרצון והמאמצים של חברות הטלפונים אשר הסבו עצמן למתן שירותי העברת נתונים, עדיין קיימות מגבלות אשר מונעות ניצול מירבי ונכון של אמצעים אלו להעברת נתונים.

עקב סיבות אלו מוקמות כיום ברחבי העולם חברות המתמחות בהתקנת מערכות תקשורת אשר מיועדות אך ורק לתקשורת מחשבים. המטרה היא לבנות רשת תקשורת אשר תהיה כלכלית יותר וכמובן עם אחוז שגיאות מינימלי (כלומר, ללא רעשים).

תקשורת נתונים (Data Communication)

עם התקדמות טכנולוגיית המחשבים נוצר מצב המאפשר למספר רב של משתמשים להשתמש במחשב אחד גדול. דבר זה ניתן לביצוע רק בעזרת מערכת תקשורת נוחה ואמינה.

בין היישומים הטיפוסיים במערכת תקשורת נמצאים: חברות גדולות בעלות סניפים רבים המרכזות את נתוני המלאי או מנגנון, מדענים במעבדות המעוניינים בשימוש במערכת מחשב מרוחקת, רשתות בתי־מלון המשתמשות במערכת הזמנת מקומות הקושרת מספר רב של בתי־מלון, וכד׳.

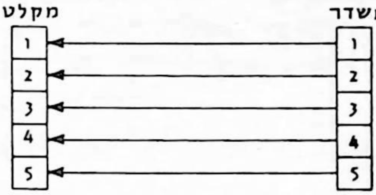
יישומים אלו ומספר רב של אחרים הגדילו את הדרישות למערכות תקשורת.

שיטות ומערכות לתקשורת נתונים

המחשבים הסיפרתיים פועלים, כזכור, על־ידי הצגה בינרית של הנתונים. גם בהעברת נתונים בין מחשבים לבין מסופים, הנתונים המועברים מיוצגים בצורה בינרית.

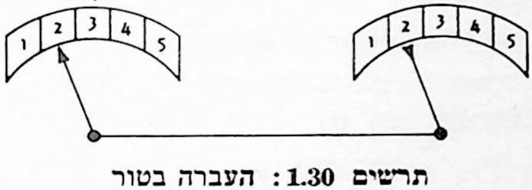
כדי להעביר תווים מוצפנים, יש לסדר את מרכיביהם בצורה שהצד המקבל יבין אותם. קיימות שתי שיטות עיקריות לבצע זאת: העברה במקביל (parallel transmission) והעברה בטור (serial transmission).

בהעברה במקביל, מועברת כל סיבית של התו במוליך נפרד, כך שכל המרכיבים מועברים בו־זמנית. לדוגמא, אם התו מורכב מחמש סיביות, יש צורך בחמישה מוליכים כדי להעביר את התו כולו. דוגמא לכך מוצגת בתרשים 1.29.



תרשים 1.29: העברה במקביל

בהעברה בטור, נשלחת כל סיבית של התו בתורה. שיטה זאת מאפשרת העברת הנתונים על גבי מוליך אחד בלבד. דוגמא לכך מוצגת בתרשים 1.30.



תרשים 1.30: העברה בטור

תקשורת נתונים והעתיד

תקשורת נתונים היא שטח חדש בתקשורת, אולם תוך זמן קצר תפס שטח זה מקום בעל חשיבות בתחום המחשבים. עובדה זאת נעוצה בעיקר בחידושים הטכנולוגיים. יחד עם זאת קיימת הדרישה בענפים שונים לשימוש במערכות תקשורת, כתוצאה מהכרה ביתרונות הנובעים מתקשורת נתונים.

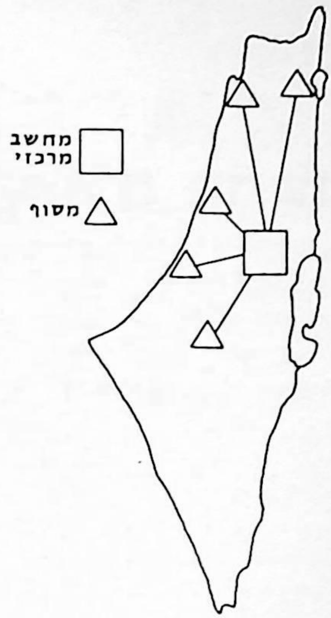
רוב הקשיים אשר היו קיימים בעבר הקרוב נפתרו למעשה וניתן כיום להתקשר למחשב מסוים מכל מקום שהוא, אם בעזרת קווים קבועים (נל״ן) או בעזרת רשת החיוג הרגילה.

יש להניח שמגמה זאת, של שימוש רב בתקשורת נתונים, תתחזק בעתיד ותהיה בעלת משקל רב בעיבוד נתונים.

.6
תקשורת מחשבים

מבוא

מערכות מחשבים המחוברות ביניהן באמצעות קווי תקשורת יוצרות רשת מחשבים (computer network). ההתפתחות המהירה בתקשורת נתונים ורשתות מחשבים חדרה לשטחים רבים — כיום נפוץ השימוש בתקשורת מחשבים בשטח העסקים, בשטחי התעשייה, המדע והמינהל הציבורי. בשנים האחרונות גדל קצב הקמת רשתות אלו, ויש להניח שמגמה זו תימשך גם בעתיד ובקצב גדול יותר.
תקשורת מחשבים מאפשרת למפעלים וארגונים גדולים לרכז את הנתונים של הסניפים המרוחקים במקום אחד מרכזי ולקבל מידע מעודכן על הנעשה בסניפים למרות המרחקים הגדולים. תקשורת מחשבים מאפשרת גישה למאגרי נתונים משותפים, תוכניות משותפות וכדומה. ניתן גם לנצל בצורה יעילה יותר את המחשבים הכלולים ברשת התקשורת על־ידי הקצאה ותזמון נכונים של העיבודים.
השימוש הנרחב בתקשורת מחשבים החל למעשה באמצע שנות החמישים, כאשר הוכח שניתן לחבר מכשירים פשוטים (כמו טלפרינטרים) למחשב, להזין את המחשב בנתונים באמצעות מכשירים אלה ולקבל חזרה את התוצאות באמצעותם.
ההתפתחות החשובה ביותר אשר איפשרה שימושים נרחבים ברשתות מחשבים היתה שיטת שיתוף־הזמנים (time sharing), שיטה המאפשרת חיבור מספר מסופים למחשב אחד מרכזי, ובעזרתה ניתן לבצע עיבודים רבים בעת ובעונה אחת. תרשים 1.31 מציג מערכת שבה מחוברים מספר מסופים למחשב אחד מרכזי והמאפשרת ביצוע מספר עיבודים בעת ובעונה אחת.



תרשים 1.31: דוגמא למערכת שתוף־זמנים

רשתות מחשבים

מאז פיתוח שיטת שיתוף־הזמנים מהווה המחשב את המרכז של רוב מערכות תקשורת הנתונים. עקב הדרישות ההולכות וגדלות לשימוש במחשב, הוכרחו רוב המשתמשים בו להגדיל או להחליף במשך הזמן את המחשבים שברשותם. זמן קצר לאחר השינוי נוצר בדרך כלל הכרח להגדיל את הקיבולת שוב. כחלוף זמן נוסף לא הספיק גם גידול זה ואז עמדו המשתמשים בפני כמה אפשרויות: להימנע מיישום שימושים נוספים, להגדיל שוב את קיבולת מערכת המחשב הנוכחית, להחליף את המחשב בגדול יותר או להתקין מחשבים נוספים ולחברם באמצעות קווי תקשורת.

רשת מחשבים מוגדרת כקבוצת מחשבים המחוברים ביניהם באמצעות קווי תקשורת, כאשר חלק מהמחשבים משמשים לעיבוד (processing systems) וחלק לפיקוח על הרשת (communications control systems). מערכות העיבוד מתחלקות לשני סוגים:

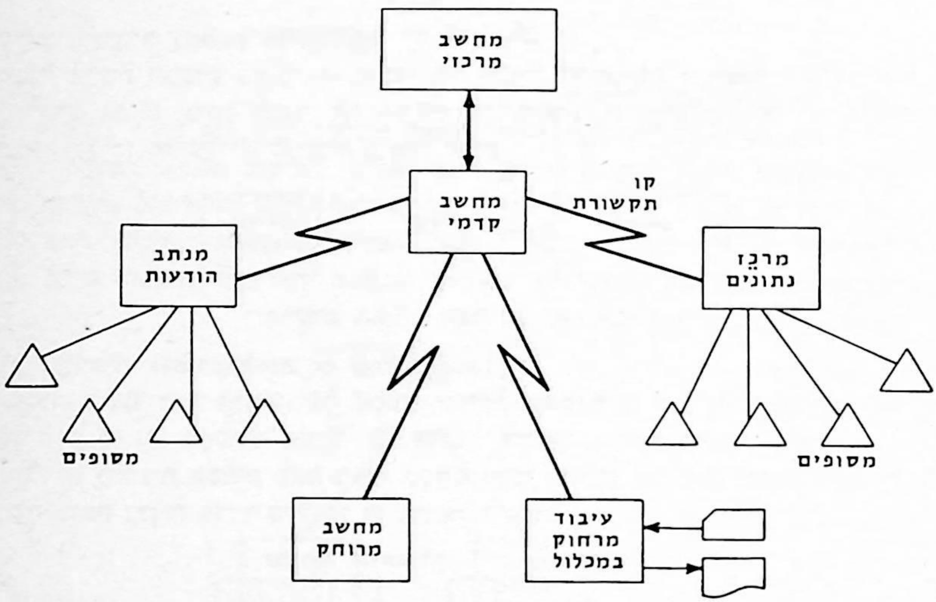
א. מחשבים מרכזיים (main-site computers, host computers).

ב. מערכות עיבוד מרוחקות (remote computing systems).

המחשבים המרכזיים ברשת מבצעים את העיבודים העיקריים, מפקחים על מאגרי הנתונים המרכזיים, ובדרך כלל גם מפקחים על פעילות הרשת. המחשבים המחוברים לרשת יכולים להתחלק ביניהם במשאבים כגון תוכניות, מאגרי נתונים ואמצעי אחסון.

מערכות עיבוד מרוחקות הם בדרך כלל מחשבים מקומיים אשר להם קשר עם המחשבים המרכזיים של הרשת. הם מבצעים חלק מהעיבודים, אשר בלעדיהם היו מתבצעים על־ידי המחשב המרכזי; כך הם מקלים את הלחץ על המחשב המרכזי. כמו כן הם מבצעים אותו חלק של העיבוד הניתן לביצוע במערכת העיבוד המרוחקת, אשר קרוב לוודאי, גם קרובה למקום כניסת הנתונים לרשת.

ברשת מחשבים, יכול כל אחד מהמחשבים המרכזיים והמרוחקים לפעול גם כמחשב מקומי. לפיכך יכולה רשת מחשבים לכלול מספר מחשבים מרכזיים, מספר מחשבים מרוחקים, מספר מחשבי פיקוח, ומספר רב של מסופים. תרשים 1.32 מציג רשת טיפוסית הכוללת את רוב מרכיבי הרשת. הסבר מפורט של מרכיבי הרשת ניתן בסוף פרק זה.



תרשים 1.32: מרכיבי רשת טיפוסית

שיתוף משאבים (Resource Sharing)

סיבות חשובות להתעניינות ההולכת וגוברת ברשתות מחשבים הן היתרון הכלכלי והנוחות המושגים בעזרת השיתוף במשאבים. על־ידי חיבור מספר מחשבים ניתן ליצור כוח חישוב משותף גדול, ניתן להשתתף במאגרי נתונים גדולים ובתוכנה מיוחדת. כל המשאבים, הנמצאים במקומות השונים אשר בהם מותקנות מערכות המחשבים, עומדים לרשות כל אחד מהמשתמשים המחוברים לרשת.

בשיתוף משאבים נכללים ההיבטים הבאים:

שיתוף בציוד (device sharing) — היכולת של מחשב הקשור לרשת להתחבר לציוד הנמצא במקומות מרוחקים, כאילו החיבור הנו לציוד המקומי.

שיתוף קבצים (file sharing) — היכולת לקרוא, לכתוב או לעדכן מאגרי נתונים הנמצאים במקום מרוחק, כאילו היו מקומיים (קובץ הנו מאגר מאורגן של נתונים).

שיתוף בתוכניות (program sharing) — היכולת של מחשב אחד להשתמש בתוכניות הנמצאות במקומות מרוחקים.

שיתוף בנתונים לתוכניות (program data sharing) — היכולת להעביר נתונים בין תוכניות מחשב המופעלות במקומות מרוחקים, כך שעבודות גדולות מפוצלות למספר עבודות קטנות המתבצעות במספר מקומות.

סוגי רשתות

נקודה לנקודה (point-to-point)

מבנה הרשת הפשוט ביותר — חיבור של יחידת קלט/פלט מרוחקת למחשב מרכזי (תרשים 1.33). ניתן לחבר את יחידת הקלט/פלט למחשב על-ידי קו תקשורת קבוע (נל"ן), או על-ידי שימוש ברשת החיוג.



תרשים 1.33 : רשת נקודה לנקודה

רב נקודתית (multidrop or multipoint)

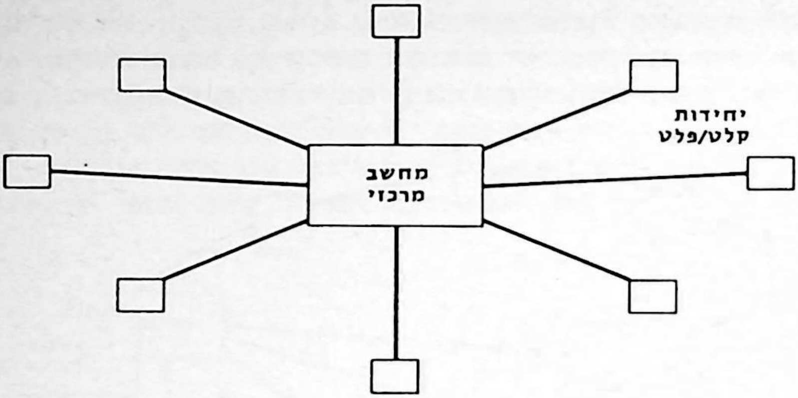
המבנה השני הוא חיבור של מספר יחידות קלט/פלט על קו תקשורת משותף (תרשים 1.34). המחשב מפקח על הרשת על-ידי משאל מחזורי (polling) של היחידות השונות הבודק באם מסוף מסוים מוכן לשלוח או לקבל נתונים מהמחשב. רשת כזאת נבנית בדרך כלל על קו תקשורת קבוע.



תרשים 1.34 : רשת רב נקודתית

מבנה ריכוזי או מבנה כוכב (centralized or star structure)

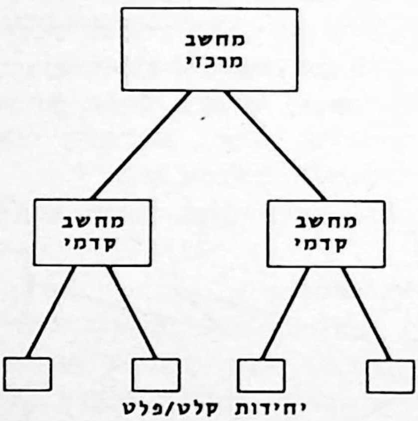
דוגמא למבנה הכוכב מוצגת בתרשים 1.35. בשיטה זאת מתקשרים כל יחידות הקלט/פלט ישירות למחשב המרכזי באמצעות קווי תקשורת נפרדים. כאשר יש להעביר הודעות בין היחידות ההיקפיות, משמש המחשב המרכזי כמנתב ההודעות. מבנה זה של רשת מאפשר פיקוח פשוט על הרשת.



תרשים 1.35 : מבנה כוכב

מבנה עץ (tree structure)

מבנה זה מקובל בעיקר בתעשייה, מאחר שבדרך כלל יש צורך בפיקוח וניהול של יישומים תעשייתיים מסוימים בזמן אמיתי (תרשים 1.36). במערכות במבנה עץ ניתן לפקח על תהליכים, לכוונם ולדווח על מצבם: מערכות גישוש מספקות למערכת הפיקוח נתונים בזמן אמיתי, המחשבים ברמת הפיקוח מתאמים את התהליכים ובמקביל מעבירים נתונים למחשב המרכזי המעבד מידע הקשור בתכנון, בקרה, ניהול וכד׳.

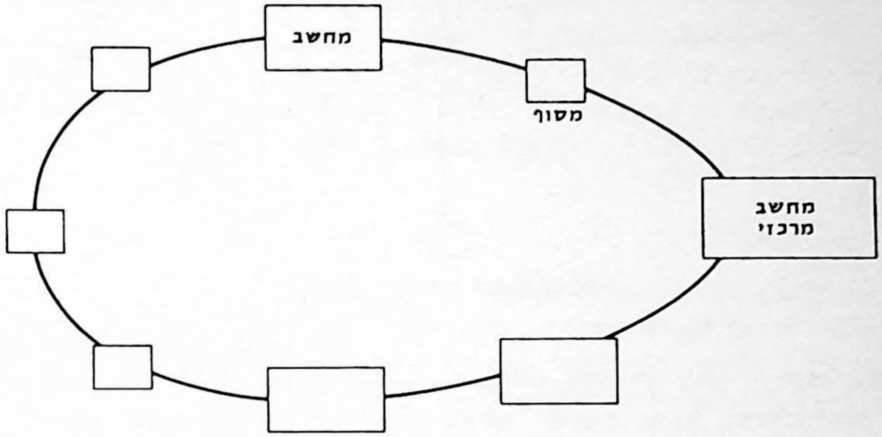


תרשים 1.36 : מבנה עץ

מבנה טבעת (loop or ring structure)

ארגונים רבים תכננו את רשת המחשבים שלהם בצורת מבנה הטבעת. מבנה זה אינו מאפשר למספר רב של תחנות מרוחקות (מסופים או מחשבים) המחוברות לה להתקשר ישירות למחשב המרכזי. בשיטה זאת מועברים הנתונים סביב התחנות (תרשים 1.37).

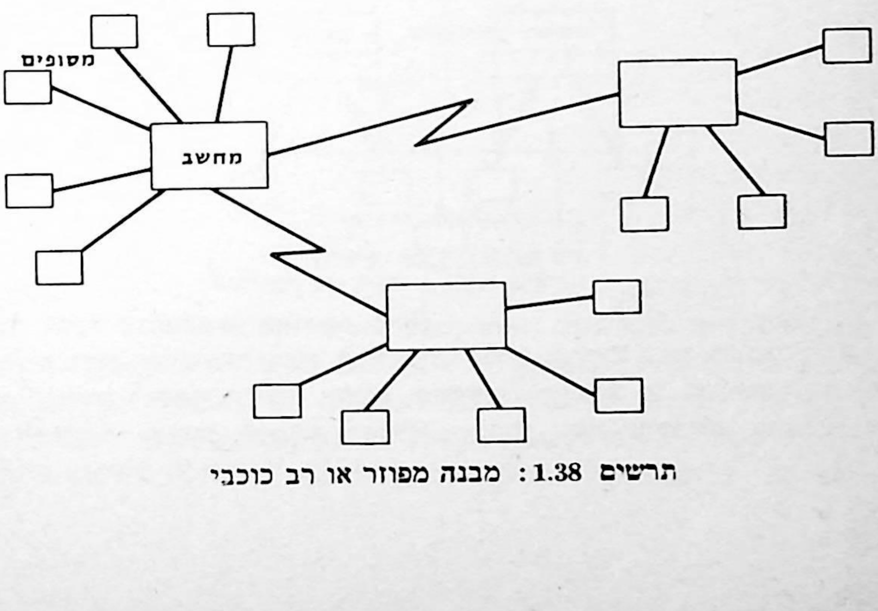
מבנה זה הוא כדאי מבחינה כלכלית כאשר התחנות נמצאות קרובות זו לזו. כאשר התחנות מרוחקות בצורה ניכרת יהיה מבנה זה יקר מדי עקב מחירם של קווי התקשורת; במקרה כזה יהיה כדאי יותר המבנה המפוזר (תרשים 1.38 ו־1.39).



תרשים 1.37: מבנה טבעת

מבנה מפוזר או רב כוכבי (distributed, multistar structure)

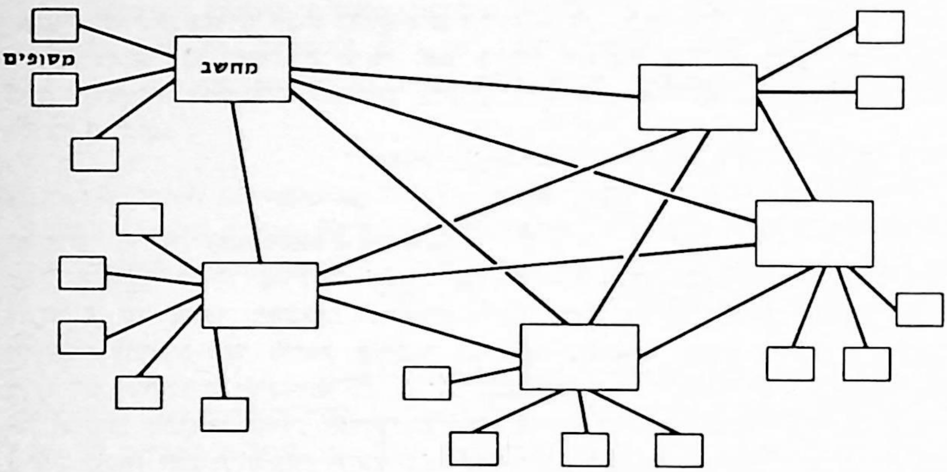
תרשים 1.38 מראה מערכת מפוזרת עם מספר תצורות כוכביות, כאשר קיימות מספר נקודות פיקוח או ניתוב, המפקחות על תנועת הנתונים ברשת.
מבנה כזה עשוי להיות חסכוני כאשר ליד כל ריכוז של מסופים נמצא מחשב שאליו מתחברים המסופים הקרובים אליו גיאוגראפית. אם רשת מפוזרת מתוכננת נכון, היא יכולה להיות אמינה יותר, מאחר שתקלות ביחידה מסוימת אינן משפיעות על היחידות האחרות ברשת.



תרשים 1.38: מבנה מפוזר או רב כוכבי

פיזור מלא (fully distributed network)

כאשר אמינותה והמשכיותה של פעולת התקשורת הנן חשובות, עדיפה מערכת בפיזור מלא (תרשים 1.39). תוספת של ערוצי העברת הנתונים ברשת זו משפרת את יעילות הרשת. בתכנון רשת כזאת יש לנתח היטב את היקף תנועת הנתונים הצפוי, זאת לשם קביעת מיקומם של מוקדי הרשת. עקב מחירן הגבוה של רשתות מסוג כזה מספר מועט מהן יושמו הלכה למעשה. קרוב לוודאי שבעתיד יגדל השימוש ברשתות כאלו.



תרשים 1.39 : רשת בעלת מבנה של פיזור מלא

ניתוב הודעות (Message Switching and Routing)

ניתוב הודעות היא שיטת הטיפול בהעברת נתונים ברשתות תקשורת. הנתונים מועברים לנקודת ביניים (מחשב מנתב), מאוחסנים לתקופת זמן, בדרך כלל קצרה מאוד, ולאחר מכן מועברים שוב לכיוון היעד.

קיימות שלוש שיטות בסיסיות בניתוב תנועת תקשורת מהמקור ליעד ובחזרה:

מתוג מעגלים (circuit switching)

ברשת עם מתוג מעגלים, קיימת מרכזייה שתפקידה ליצור קשר ישיר בין מסוף או מחשב למשנהו. לאחר יצירת הקשר, המסופים או המחשבים (המקור והיעד) מעבירים את המידע ביניהם. כאשר מסתיימת העברת ההודעה, מנתקת המרכזיה את המעגל ומחזירה את המערכת למצב נכונות לקראת ההתקשרות הבאה. רשת הטלפונים הרגילה פועלת בצורה דומה.

ניתוב הודעות (message switching)

בשיטה זאת, כל הודעה הנשלחת לרשת מנותבת כיחידה אחת ליעד. הודעות נוספות בין אותם מקור ויעד יכולות להישלח בנתיב אחר לגמרי. כך שלמעשה הקשר בין מקור מסוים הוא לא קבוע כמו בשיטה הקודמת, אלא נקבע בהתאם

למצב הרשת בזמן שבו נוצר הצורך לשלוח הודעה. כל הודעה נשלחת עם כתובת היעד. כיוון שלעתים יכול להיווצר מצב שאחת מתחנות היעד שאליה נשלחת ההודעה עסוקה, מאוחסנת ההודעה בתחנת ביניים עד אשר תתפנה תחנת היעד. לפיכך מכונה שיטה זאת גם "אחסן ושלח" ("store and forward" system).

ניתוב צרורות (packet switching)
שיטה זאת דומה במהותה לשיטת ניתוב ההודעות, אלא שההודעות ארוכות מפוצלות במקור לחלקים באורך קבוע (הנקראים צרורות), ואז נשלח כל צרור בנפרד בדרך הטובה ביותר. כן מאפשרת שיטה זאת משלוח הודעות ארוכות בזמן קצר יותר, כיוון שההודעה מפוצלת לחלקים קטנים, הנשלחים בו זמנית בנתיבים שונים לאורך הרשת.

שימושי הרשת (Network Functions)
כפי שהוסבר לעיל, מורכבת רשת תקשורת מחשבים מחיבור באמצעות קווי תקשורת של מספר מחשבים ומסופים. רשת כזאת יכולה להיות בנויה ממסוף המחובר למחשב ועד לרשת מורכבת הכוללת מסופים, מרכזי נתונים, מחשבים מרוחקים ומחשבים מרכזיים. כל מחשב במערכת כזאת נבחר לפי יכולתו לבצע את תפקידו במקומו הפיזי. המחשבים השונים יכולים להיות בעלי מהירות פעולה שונה, שפות תכנות שונות ומאפיינים אחרים שונים. החשוב מכל שיהיו יעילים בביצוע התפקידים שנועדו להם.

עיבוד מרכזי (host processing)
מערכת העיבוד המרכזית יכולה להיות מחשב מרכזי גדול במערכת שיתוף־הזמנים הטיפוסית. היא יכולה לקלוט ולעבד קבוצה כללית של דרישות ממספר רב של התקנים מקומיים או מרוחקים כולל מחשבים ומסופים. המחשב המרכזי הנו בעל זיכרון פנימי גדול ומכלול רחב של ציוד היקפי, הכולל, בין השאר, תקליטים מגנטיים המאפשרים אחסון כמות גדולה של נתונים ומשמשים כמרכז נתונים לכל הרשת. מערכת התוכנה של המחשב המרכזי כוללת מערכת הפעלה מתוחכמת, הנתמכת על־ידי מערכת תוכנה לתקשורת המתוכננת במיוחד עבור הפעלת הרשת ומפקחת על פעילות הרשת.
בנוסף לכך, המחשב המרכזי מבצע חלק עיקרי מהעיבודים הנדרשים על־ידי המשתמשים ברשת.

עיבוד קדמי בתקשורת (front-end communications processing)
שיטה זאת מאפשרת שחרור המחשב המרכזי מטיפול במערכת התקשורת. הפיקוח על התקשורת נעשה על־ידי מחשב "קדמי" שזה תפקידו העיקרי: בתפקיד זה משמשים מחשבים זעירים, המותקנים ברשת בין המסופים למחשב המרכזי. שיטה

זאת היא יותר כלכלית, מכיוון שהמחשב המרכזי שהנו בדרך כלל יקר פי כמה ממחשב זעיר, משוחרר מטיפול בבעיות התקשורת.

ריכוז מרוחק (remote concentration)
בשיטה זאת מחוברים מספר מסופים בעלי מהירות העברת נתונים קטנה יחסית למרכז נתונים הקשור למחשב עם קו תקשורת מהיר יותר. מרכז הנתונים (data concentrator) אוסף נתונים ממספר מסופים איטיים ומעבירם הלאה במהירות גבוהה יותר. לביצוע תפקיד זה ניתן להשתמש גם כאן במחשבים זעירים. ריכוז מרוחק מאפשר הקטנת עלות הקווים ברשת התקשורת.

הזנת עבודות מרחוק (remote job entry — RJE)
קיימים מסופים המאפשרים עיבוד במכלול ממרחק (remote batch processing). בשיטה זאת מועברת העבודה (התוכנית והנתונים) כולה למחשב המרכזי ומצטרפת לתור העבודות הממתינות לעיבוד. לאחר ביצוע העבודה נשלחות התוצאות חזרה לנקודה שממנה הוזנה לרשת.

מערכת מחשב מרוחקת (remote computer system)
שלא כשיטת העיבוד מרחוק במכלול, מסוגל המחשב המרוחק לבצע חלק מהעיבודים בעצמו. רק באותם מקרים שלא ניתן לבצע בהם את העיבוד במחשב המרוחק, נשלחת העבודה לביצוע למחשב המרכזי.

7.

ארגון קבצים (File Organization)

כל ארגון נוהג לרכז את הנתונים הדרושים לו לצורך תפעול, ניהול וקבלת החלטות. השימוש במערכת המחשב לטיפול במאגרי הנתונים של הארגון הנו בעל יתרונות רבים: מערכת המחשב מעמידה בפני המשתמש מגוון של אמצעי אחסון לנתונים, שיטות לארגון הנתונים על גבי יחידות האחסון, ומאפשרת לטפל בנתונים ולעבדם ביעילות. פרק זה דן בשיטות האחסון והעיבוד של מאגרי הנתונים במערכת המחשב.

הגדרות יסוד

תו (character): יחידה בסיסית של נתונים, המקבילה לאות או לסיפרה בודדת.

שדה (field): מספר תווים המהווים נתון אחד. לדוגמא, מספר תעודת זהות או שם.

רשומה (record): קבוצת שדות בעלי קשר לוגי ביניהם. לדוגמא, כל השדות המתייחסים לעובד מסויים או לפריט במלאי.

קובץ (file): אוסף כל הרשומות השייכות לנושא מסויים. לדוגמא, אוסף הרשומות של העובדים במפעל. בדרך כלל כל הרשומות בקובץ בעלות מבנה זהה.

קובץ־אב (master file): קובץ ראשי או עיקרי המכיל את הנתונים הקבועים והמשתנים השייכים לנושא מסויים. לדוגמא, קובץ־אב עובדים יכיל את פרטיהם של כל העובדים: מספר תעודת זהות, שם, כתובת, דירוג מקצועי, מחלקה, וכו׳.

קובץ תנועות (transactions file): קובץ המכיל תנועות שינוי או חידוש לקובץ־האב, המשמש לעדכון קובץ־האב. לדוגמא, שינוי כתובת של אחד העובדים, מספר שעות עבודה בחודש של עובד מסויים, וכו׳.

קובץ היסטורי (historical file): קובץ־אב ישן, אשר נשמר לצורך גיבוי לקובץ־האב המעודכן, ומאפשר לשחזרו במקרה של תקלה.

רב־קובץ (data base): קובץ או אוסף של קבצים הכוללים את כל הנתונים הקשורים ישירות או בעקיפין לנושא מסויים. כל נתון מופיע פעם אחת בלבד.

מפתח (key): שדה ברשומה המזהה אותה ומאפיין אותה מרשומות אחרות.

תכונות של קבצים

פעילות: תדירות השינויים בקובץ; דהיינו — תכיפות הביטול, העדכון וההוספה של רשומות לקובץ. לדוגמא, קובץ־אב מלאי חלקי חילוף במוסך למכוניות הוא בעל פעילות רבה, ואילו קובץ־אב חברי־כנסת במדינת ישראל הנו בעל פעילות נמוכה.

העדכניות הדרושה: משך הזמן המקסימלי אשר יכול לעבור בין השינוי שחל במציאות לבין ביצוע העדכון הנדרש בקובץ כתוצאה מהשינוי. העדכניות הדרושה מוכתבת על־ידי המשתמש בקובץ.

אחוז רשומות פעילות: מספר הרשומות אשר חל בהן שינוי בכל תקופה יחסית למספר הרשומות הכללי בקובץ. ייתכן קובץ בו הפעילות שווה בממוצע לכל הרשומות; ייתכן קובץ אחר בו רוב השינויים מתבצעים במספר קטן יחסית של רשומות.

גודל: המקום הדרוש לאחסון הקובץ, מבחינת מספר התווים. גודל זה נקבע על־ידי מספר התווים בכל רשומה ומספר הרשומות בקובץ.

פוטנציאל הגידול: שיעור הגידול הצפוי של הקובץ בעתיד.

תכונות של ציוד האחסון

יחידות האחסון במערכת המחשב מתחלקות לשני סוגים עיקריים, בהתאם לאפשרויות ארגון הקבצים המאוחסנים עליהן.

יחידות סידרתיות: ביחידות אלה הרשומות מסודרות זו אחר זו לפי סדר (עולה או יורד) של מפתח המיון. קריאה או כתיבה של נתונים מחייבת מעבר סידרתי על פני כל הרשומות עד לרשומה הרצויה.

יחידות סידרתיות הן הסרט המגנטי, כרטיסים מנוקבים וסרט נייר מנוקב. יתרונן של יחידות אלה במחירן הזול יחסית.

יחידות גישה ישירה: ביחידות אלה לכל רשומה יש "כתובת" המאפשרת לאתר את הרשומה ישירות, מבלי לעבור על הרשומות האחרות.

יחידות גישה ישירה הן התקליט המגנטי, התוף המגנטי וה־CRAM. יחידות אלה יקרות בהרבה בהשוואה ליחידות הסידרתיות.

שיטות ארגון קבצים

ארגון הרשומות בקובץ צריך להיעשות בצורה שתאפשר עיבוד ושימוש יעיל בנתונים הנמצאים בקובץ. שיטת ארגון הקבצים המתאימה ליישום מסוים תלויה בתכונות הקבצים וביישומים עבורם נדרשים הקבצים; יש לנתח גורמים אלה לצורך תכנון נכון של ארגון הקבצים. ארגון יעיל של הקבצים יכול להשפיע בצורה משמעותית על יעילות כל מערכת המחשב.

קיימים שלושה מבנים בסיסיים לארגון קבצים:

א. קובץ סידרתי (sequential file).

ב. קובץ טבלתי (index-sequential file).

ג. קובץ ישיר/אקראי (random/direct access file).

א. ארגון קבצים סידרתי

קבצים סידרתיים מאוחסנים בדרך כלל על גבי סרטים מגנטיים, אך גם כרטיסים מנוקבים, סרט נייר מנוקב, תקליט מגנטי או תוף מגנטי יכולים לשמש לאחסון קבצים סידרתיים.

בשיטה זו מסודרות הרשומות בקובץ לפי מפתח מיון בסדר עולה או יורד. בזמן עיבוד (קריאה, כתיבה, הוספה, ביטול או שינוי של רשומות בקובץ) קובץ סידרתי, נעשה העיבוד לכל רשומה לפי סדר הופעתה בקובץ; אין אפשרות לבצע עיבוד לרשומה לפני שבוצע העיבוד הנדרש לכל הרשומות שלפניה. בעיבוד הדורש הוספה או ביטול של רשומות מקובץ סידרתי יש להעתיק את הקובץ מחדש: מאחר שהרשומות בקובץ הקיים כתובות אחת אחרי השניה, אין מקום על גבי מתקן האחסון להוסיף רשומה בין שתי רשומות קיימות, ומאותה סיבה אין אפשרות לבטל רשומה הכתובה על גבי הקובץ מבלי להשאיר חלל ריק במקום בו הרשומה היתה כתובה. לכן, כאשר בזמן העיבוד יש לבטל או להוסיף רשומות לקובץ, הקובץ הקיים (הישן) נקרא לצורך קבלת הנתונים, וקובץ חדש נכתב — המכיל את הנתונים החדשים, וכולל את הרשומות החדשות שנוספו, ובו צומצמו הרווחים של הרשומות שבוטלו.

מאחר שקריאה וכתיבה של הרשומות בקובץ הסידרתי נעשית לפי סדר הופעת הרשומות בקובץ, גם עדכון הקובץ חייב להיעשות בסדר זה, ולכן על קובץ התנועות — המכיל רשומות עם שינויים לקובץ — להיות ממויין באותו סדר מיון כמיון הקובץ אותו יש לעדכן.

לדוגמא נראה מהלך עדכון של קובץ סידרתי (תרשים 1.40):

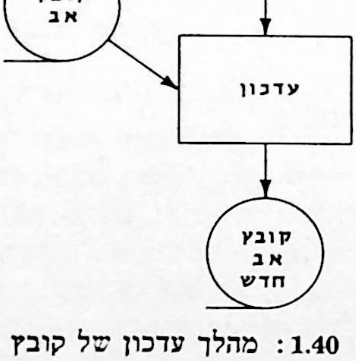
הקלט למהלך העדכון הוא קובץ־אב וקובץ תנועות, שניהם ממויינים לפי אותו סדר מיון. הפלט ממהלך העדכון הוא קובץ־אב חדש, בו נעשו כל השינויים הדרושים, והוא ממויין בסדר המיון של הקבצים האחרים (קבצי הקלט).

תוכנית העדכון קוראת רשומה אחת מכל אחד מהקבצים, ובודקת את שדות המפתח בשתי הרשומות:

— אם מפתח הרשומה שנקראה מקובץ־אב נמוך ממפתח הרשומה שנקראה מקובץ התנועות — פירוש הדבר שאין עדכון לרשומה שנקראה מקובץ־האב, והיא מועתקת ללא שינוי אל קובץ־האב החדש; אחרי כתיבת הרשומה בקובץ־האב החדש, התוכנית קוראת את הרשומה הבאה מקובץ־האב, וחוזרת לבדיקת המפתחות.

— אם מפתח הרשומה שנקראה מקובץ־האב זהה למפתח הרשומה שנקראה מקובץ התנועות — פירוש הדבר שיש לבצע עדכון לרשומה מקובץ־האב בהתאם לנתונים שברשומה שנקראה מקובץ התנועות. לאחר עדכון הרשומה, נכתבת הרשומה המעודכנת על גבי קובץ־האב החדש. כאשר התנועה שנקראה מקובץ התנועות פירושה ביטול הרשומה מקובץ־האב, לא תכתב רשומה על גבי קובץ־האב החדש. בגמר העדכון התוכנית קוראת את הרשומה הבאה מכל אחד משני קבצי הקלט, וחוזרת לבדיקת המפתחות.

— אם מפתח הרשומה שנקראה מקובץ־האב גדול ממפתח הרשומה שנקראה מקובץ התנועות — פירוש הדבר שיש להוסיף רשומה חדשה לקובץ־האב החדש, בהתאם לנתונים ברשומה שנקראה מקובץ התנועות. התוכנית כותבת רשומה זו בקובץ־האב החדש, ולאחר מכן קוראת את הרשומה הבאה מקובץ התנועות, וחוזרת לבדיקת המפתחות.



תרשים 1.40: מהלך עדכון של קובץ סידרתי

מתיאור מהלך העדכון נובע שאם מספר התנועות קטן, חלק גדול מהכתיבה תהיה פעולת העתקה גרידא.

יתרונות ארגון קבצים סידרתי — חסכוני במקום, פשוט, וניתן להפעלה באמצעי אחסון זולים. חסרונותיו — אין אפשרות לקרוא רשומה מסוימת מתוך הקובץ מבלי לקרוא את כל קודמותיה, ולכן העיבוד דורש זמן ניכר. אין אפשרות להוסיף או

לבטל רשומות מקובץ קיים מבלי להעתיק אותו, ולכן במהלך העדכון יש לכתוב מחדש את הקובץ.

ג. ארגון קבצים טבלתי

קבצים מסוג זה מאוחסנים בדרך כלל על גבי תקליטים מגנטיים, ולעתים גם על גבי תוף מגנטי. שיטה דומה משמשת לארגון קבצים על גבי ה־CRAM.

בזמן בניית קובץ טבלתי, ממויינות הרשומות בסדר עולה לפי מפתח המיון, והן נכתבות על גבי התקליט בסדר זה. בתחילת התקליט נמצא מדריך הגלילים (cylinder index) המכיל את מספר המפתח של הרשומה הגבוהה ביותר שנמצאת על כל גליל של התקליט. בנוסף לטבלה זו, במסילה העליונה בכל גליל נמצא מדריך המסילות (track index) המכיל את מספר המפתח של הרשומה הגבוהה ביותר שנמצאת על כל מסילה באותו גליל.

בזמן עדכון הקובץ, חיפוש רשומה מסוימת שהמפתח שלה נתון מתבצע בשלבים הבאים:

1. התוכנית סורקת את מדריך הגלילים, ומוצאת באיזה גליל כתובה הרשומה הרצויה.
2. כשנמצא הגליל בו מופיעה הרשומה הרצויה, סורקת התוכנית את מדריך המסילות של אותו גליל, ומוצאת באיזו מסילה כתובה הרשומה הרצויה.
3. כשנמצאת המסילה בגליל, סורקת התוכנית את המסילה, ומוצאת בה את הרשומה הרצויה.

אם לצורך עדכון רשומה יש לשנות את הנתונים ברשומה — נכתבים הנתונים החדשים באותו מקום בו נמצאה הרשומה בקובץ. בעית ביטול רשומות קיימות והוספת רשומות חדשות לקובץ מסובכת יותר, מכיוון שהרשומות מסודרות זו אחר זו ללא רווחים ביניהן.

ביטול רשומה מהקובץ נעשה על־ידי סימון הרשומה כמבוטלת, והשארתה במקומה בקובץ עד לארגון מחדש של הקובץ (ראה להלן).

הוספת רשומה חדשה לקובץ נעשה באופן הבא: התוכנית מחפשת בעזרת מדריך הגלילים ומדריך המסילות את המקום בו צריכה הרשומה החדשה להיות על גבי הקובץ, בהתאם למפתח הרשומה וסדר מיון הקובץ. אם במסילה שנמצאה קיים מקום פנוי המספיק לכתיבת הרשומה החדשה — התוכנית תזיז את הרשומות בעלות מפתח גבוה יותר הנמצאות על אותה מסילה כדי לפנות את המקום המיועד לרשומה החדשה, ואז תכתוב אותה במקומה, כך שמיקום הרשומות הישנות והרשומה החדשה יהיה בהתאם לסדר המיון של הקובץ.

אם במסילה שנמצאה אין מקום פנוי עבור הרשומה החדשה, התוכנית תזיז את הרשומות בעלות המפתח הגבוה יותר הנמצאות על אותה מסילה כדי לפנות את המקום המיועד לרשומה החדשה. כתוצאה מהזזה זו, הרשומה האחרונה על גבי המסילה תהיה מחוסרת־מקום על המסילה. במקום שפונה על־ידי הזזת הרשומות

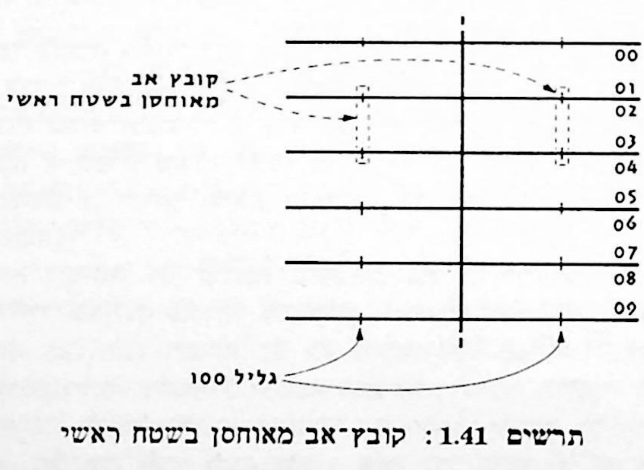
נכתבת הרשומה החדשה. הרשומה האחרונה, שהוצאה מהמסילה, תכתב על־ידי התוכנית על גבי התקליט בשטח מיוחד — הנקרא שטח גלישה.

כאשר התוכנית תחפש רשומה מסוימת — לאחר סריקת המסילה שנמצאה לפי מדריך הגלילים ומדריך המסילות — סורקת התוכנית גם את שטח הגלישה של אותה מסילה, כאילו שטח הגלישה הנו המשך של המסילה.

בדרך זו מיקום הרשומות על גבי הקובץ הנו לכאורה לפי סדר המיון של הקובץ, בעוד שלמעשה, מחולקות הרשומות בין שני שטחי הקובץ — השטח הראשי (prime data area) ושטח הגלישה (overflow area).

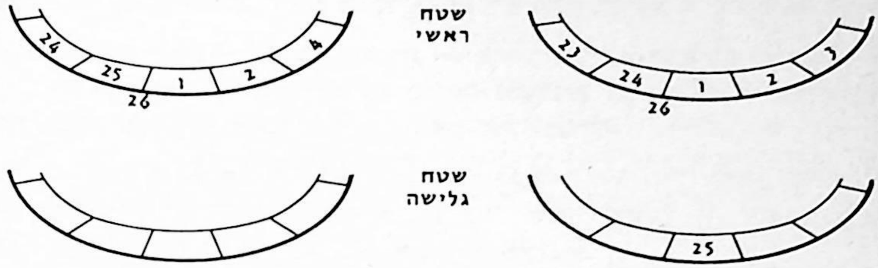
לאחר כמה עיבודים בהם נוספות רשומות רבות לקובץ — ייתכן שמספר הרשומות בשטח הגלישה יהיה גדול, דבר הגוזל זמן בחיפוש אחר רשומות בקובץ. לכן, מפעם לפעם מאורגן הקובץ מחדש. הארגון מחדש פירושו מיון כל הרשומות השייכות לקובץ (הרשומות הנמצאות בשטח הראשי והרשומות הנמצאות בשטח הגלישה) לפי סדר המיון של הקובץ, סילוק הרשומות שסומנו כמבוטלות, וכתיבת הקובץ מחדש. כתוצאה מהארגון מחדש יהיו כל הרשומות מסודרות בשטח הראשי לפי סדר המיון, שטח הגלישה יהיה ריק ומוכן לקלוט רשומות חדשות, ומדריכי הגלילים והמסילות יהיו מעודכנים בהתאם לסדר החדש של הרשומות בקובץ.

דוגמא: קובץ־אב של העובדים במפעל מכיל רשומה אחת לכל עובד, הכוללת את מספרו, שמו ופרטים אחרים של העובד. הקובץ ממויין בסדר עולה לפי מספר העובד (מפתח הרשומה) ומאוחסן על גבי תקליט. השטח הראשי של הקובץ נמצא בגליל 100 במסילות 1, 2, 3 ו־4 (תרשים 1.41).



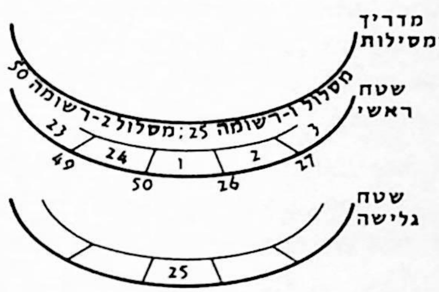
תרשים 1.41: קובץ־אב מאוחסן בשטח ראשי

בתרשים 1.42 א. ניתן לראות את תוכן השטח הראשי ושטח הגלישה כאשר הקובץ נוצר. תרשים 1.42 ב. מדגים את השימוש בשטח הגלישה כתוצאה מהכנסת רשומה חדשה של עובד מספר 3. בדוגמא זו, הועברה רשומה 25 לשטח הגלישה.



תרשים 1.42, (א, ב): מבנה הקובץ לפני ואחרי הוספת רשומה ד

תרשים 1.43 מראה את מדריך המסילות של קובץ העובדים המתואר בדוגמא. כזכור, מדריך המסילות שומר את מפתח הרשומה הגבוהה ביותר בכל מסילה.



תרשים 1.43: מדריך המסילות

בדוגמא זו, כדי לקרוא את רשומה 4 למשל, קוראת התוכנית את מדריך הגלילים, המפנה אותה לגליל 100. בגליל 100 קוראת התוכנית את מדריך המסילות, המראה שהמפתח הגבוה ביותר במסילה 1 הוא 25 (רשומה 25 נמצאת בשטח הגלישה). מכיוון שרשומה 4 בעלת מפתח הקטן מרשומה 25, יודעת התוכנית שרשומה 4 נמצאת על מסילה 1, לפיכך תקרא התוכנית את הרשומות ממסילה 1, וביניהן תמצא את רשומה 4.

כדי לקרוא את רשומה 27, קוראת התוכנית את מדריך הגלילים ואת מדריך המסילות. על־ידי בדיקת מדריך המסילות, יודעת התוכנית שהמפתח הגבוה במסילה 1 הוא 25, ולכן רשומה 27 לא תימצא על מסילה 1. המפתח הגבוה במסילה 2 הוא 50, ולכן רשומה 27 תימצא במסילה 2.

יתרונה העיקרי של שיטת הארגון הטבלתי הוא שניתן לגשת לכל רשומה בקובץ בצורה ישירה. כתוצאה מכך נובע יתרון אחר של שיטה זו והוא שאין צורך למיין את התנועות הבאות לעדכן את הקובץ. יתרונות אחרים של שיטה זו הם: הוספת רשומות לקובץ איננה מחייבת כתיבת הקובץ מחדש, וגישה מהירה וישירה לנתונים בקובץ; תכונות אלה מאפשרות לעבוד בשיטת העיבוד הישיר (on-line)

בעזרת קבצים כאלה. יתרון נוסף של השיטה הנו שאפשר לעבוד עם קבצים טבלתיים כאילו היו קבצים סידרתיים.
חסרונותיה של שיטה זו הם: התכנות מסובך יותר מאשר בשיטת הקבצים הסידרתיים, הצורך בארגון מחדש של הקבצים מפעם לפעם, והמצאותן של הרשו־מות המבוטלות על גבי הקובץ עד לארגון מחדש, דבר הגורם לבזבוז במקום. בזבוז נוסף נובע מהצורך בשמירת מקום לשטח הגלישה. חסרון נוסף הנו בכך שעדכון הקובץ נעשה על גבי הקובץ הקיים (כלומר — ללא העתקתו), וכתוצאה מכך אין קובץ היסטורי המשמש לגיבוי לקובץ החדש. קבצים טבלתיים מחייבים שימוש באמצעי אחסון יקרים — תקליט מגנטי או תוף מגנטי.

ג. ארגון קבצים ישיר/אקראי

בשיטת הארגון הישיר מאוחסנים הקבצים על גבי תקליט מגנטי או על גבי תוף מגנטי.

בשיטה זו לכל רשומה בקובץ כתובת הקובעת את מיקומה על גבי התקליט: הכתובת של כל רשומה מבוססת על מפתח הרשומה, אך איננה זהה למפתח, ולכן הרשומות אינן בהכרח מסודרות לפי סדר המיון של המפתחות. חשוב להדגיש שמיקום הרשומה על גבי התקליט ניתן לקביעה באמצעות נוסחה המבוססת על מפתח הרשומה.

בזמן עיבוד קובץ ישיר, כדי למצוא רשומה מסוימת שהמפתח שלה נתון, מתרגמת התוכנית בעזרת הנוסחה הנתונה את מפתח הרשומה לכתובת על התקליט, ובעזרת כתובת זו יודעת התוכנית להגיע לרשומה הרצויה.

שינוי נתונים ברשומה נעשה על־ידי כתיבת הנתונים החדשים ברשומה הקיימת. ביטול רשומה נעשה על־ידי הוצאתה מהקובץ, כך שמקומה נשאר ריק, והוספת רשומה נעשה על־ידי כתיבת הרשומה החדשה למקומה בקובץ.

בשיטה זו קיים בזבוז ניכר של מקומות בקובץ, מאחר שנשמר מקום על גבי התקליט לכל רשומה אפשרית — אם היא בשימוש ואם לא. קיימות שיטות מתוחכמות לחישוב הכתובות מתוך המפתח כך שמספר המקומות המבוזבזים על גבי התקליט יהיה מינימלי, אך במקרה כזה קיים סיכון שנוסחה כזו תיתן כתובת זהה לשני מפתחות שונים; קיימות שיטות להתגבר גם על בעיה זו.

דוגמא: בקובץ עובדים במפעל מפתח הרשומה הנו מספר העובד. מספרי העובדים במפעל נמצאים בין 500 ל־999.

שיטה אפשרית לקביעת כתובת כל רשומה על גבי התקליט הנה השיטה הבאה: הסיפרה השמאלית במספר העובד תציין את מספר הגליל; הסיפרה האמצעית תציין את מספר המסילה בגליל, והסיפרה הימנית במספר העובד תציין את מקום הרשומה במסילה. למשל, הרשומה המתאימה לעובד שמספרו 500 תמצא בגליל 5, במסילה 0, והיא תהיה הרשומה הראשונה (שמספרה 0) על גבי המסילה. הרשומה של עובד 501 תמצא אף היא בגליל 5, במסילה 0, ותהיה הרשומה השניה

(שמספרה 1) על גבי המסילה. הרשומה המתאימה לעובד 728 תמצא בגליל 7, במסילה 2, במקום התשיעי במסילה.
הנוסחה לקביעת הכתובות המוצגת בדוגמא איננה השיטה היחידה האפשרית, אך מבין השיטות היא הפשוטה והבזבזנית ביותר; אם במפעל יש למשל, רק 200 עובדים (שמספריהם, כאמור, בין 500 ל־999), הרי שחלק ניכר מהמקום שהוקצה לקובץ ישאר ריק: אם אין עובד שמספרו 659, אז המקום העשירי במסילה 5 בגליל 6 יהיה ריק. אם הרשומה של כל עובד קצרה, כך שניתן, למשל, להכניס 30 רשומות לכל מסילה, השיטה המוצגת לא תנצל את אורכן הקצר של הרשומות, ולא תאפשר לכתוב יותר מ־10 רשומות במסילה. בשיטה זו גם אין אפשרות לבחור את המקום על גבי התקליט בו ימצא הקובץ — הנוסחה קובעת שהקובץ ישתרע על גלילים 5, 6, 7, 8 ו־9. אם גלילים אלה תפוסים על־ידי קובץ אחר, יש צורך לקבוע נוסחה מתוחכמת יותר אשר תמקם את הקובץ במקום הפנוי על גבי התקליט.
כאמור, קיימות שיטות מתוחכמות לקביעת נוסחאות לתרגום מפתח הרשומה לכתובת על גבי התקליט. נוסחאות אלה מתחשבות בגודל הרשומות, במספר הרשומות שאפשר לכתוב על כל מסילה, במספר המפתחות בהם יעשה שימוש מתוך המפתחות האפשריים, ובמקומות הפנויים על גבי התקליט המגנטי.
יתרונות ארגון הקבצים הישיר — גישה ישירה ומהירה מאד לכל רשומה בקובץ. בגלל אפשרות הגישה הישירה — אין צורך למיין את התנועות הבאות לעדכן את הקובץ. הוספת רשומות לקובץ וביטול רשומות מהקובץ הנו פשוט ואינו דורש העתקה או ארגון מחדש של הקובץ. תכונות אלה של הארגון הישיר מאפשרות לעבוד עם קבצים כאלה בשיטת העיבוד הישיר (on־line) ביעילות מירבית.
חסרונותיה של שיטה זו הם הישארות שטחים ריקים על גבי התקליט המגנטי, דבר הגורם לניצולת נמוכה של הציוד, התכנות מסובך יותר, וחסרונו של קובץ היסטורי לגיבוי לקובץ המעודכן — כתוצאה מעדכון הקובץ ללא העתקתו.

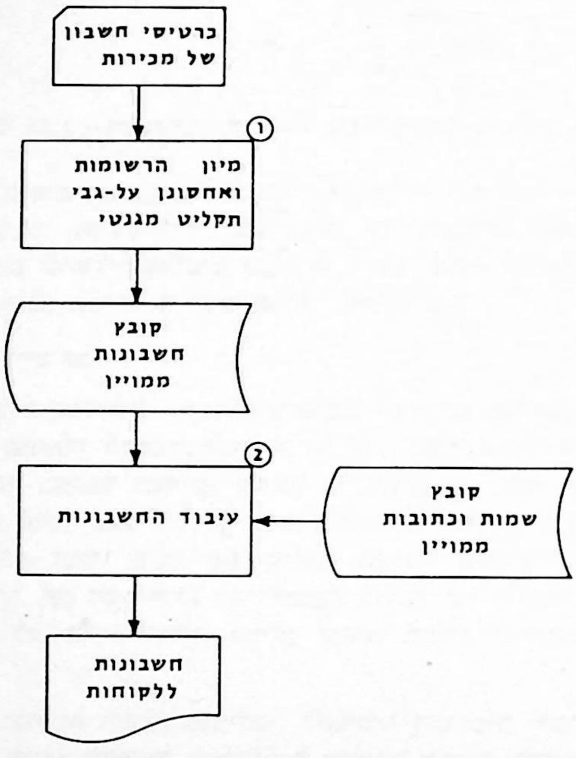
בחירת שיטת הארגון המתאימה

כאמור לעיל, בחירה נכונה של שיטת ארגון הקובץ יכולה להשפיע בצורה משמעו־תית על יעילות כל מערכת המחשב. הבעיה היא שאין כללים חד־משמעיים לקביעת שיטת ארגון הקובץ בכל יישום ויישום אפשרי. ברוב המקרים מתאימות כמה שיטות ארגון לביצוע יישום מסויים, והבחירה בין השיטות האפשריות חייבת להתבסס על מספר גורמים. השיקולים המשפיעים על בחירה זו הנם רבים ולעתים מנוגדים.
להלן נראה דוגמא בה אותו עיבוד בדיוק מתבצע בשתי שיטות שונות של ארגון קבצים. בדוגמא זו, כרטיסי המכירות של מפעל הנם קלט לעיבוד יחד עם קובץ הלקוחות. התוכנית קוראת את כרטיסי החשבונות, מחפשת את הלקוח בקובץ הלקוחות, ומכינה חשבון עבור הלקוח. כרטיסי החשבון המשמשים קלט לתוכנית אינם ממויינים.

כאשר קובץ הלקוחות הנו סידרתי — קיימים שני שלבים בתוכנית (תרשים 1.44):

1. התוכנית קוראת את כרטיסי החשבון, ממיינת אותם בהתאם למיון קובץ הלקוחות, ומאחסנת אותם על גבי תקליט מגנטי במבנה של קובץ סידרתי.

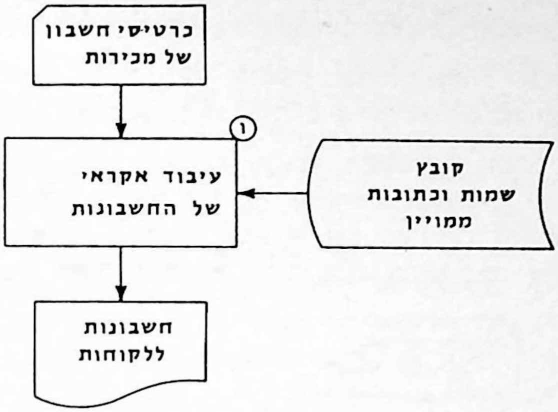
2. התוכנית קוראת את הרשומות מקובץ החשבונות הממויין ומקובץ שמות וכתובות הממויין (תהליך הקריאה והשוואת המפתחות מוסבר בסעיף הדן בקבצים סידרתיים). התוכנית מחשבת את הסכומים לתשלום, ומדפיסה את החשבון ללקוח, וממשיכה לקרוא את הרשומה הבאה בשני הקבצים.



תרשים 1.44: הכנת חשבונות — קובץ סידרתי

כאשר קובץ הלקוחות הנו קובץ טבלתי או ישיר/אקראי — קיים רק שלב אחד בעיבוד (תרשים 1.45):

התוכנית קוראת את כרטיסי החשבון, מחפשת את הרשומה המתאימה ללקוח בקובץ השמות והכתובות (תהליך חיפוש הרשומה תואר בסעיפים הדנים בקובץ הטבלתי ובקובץ הישיר/אקראי), וכאשר נמצאה הרשומה מחשבת התוכנית את הסכומים לתשלום ומדפיסה את החשבון ללקוח, ולאחר מכן ממשיכה לקרוא את כרטיס החשבון הבא.



תרשים 1.45: הכנת חשבונות — קובץ טבלתי או ישיר/אקראי

כאשר קובץ הלקוחות הנו טבלתי או ישיר/אקראי — אין צורך למיין את כרטיסי החשבונות, והעיבוד מבוצע כולו בצעד אחד, כך שהיתרון לעומת קובץ סידרתי ברור. אך קיימים שיקולים נוספים בבחירת שיטת ארגון הקובץ, ולכן אי אפשר לקבוע שתמיד ארגון טבלתי או ישיר/אקראי יעילים יותר.

השיקולים העיקריים הם:

א. **אחוז הרשומות הפעילות** — אחוז הרשומות הפעילות מתייחס למספר הרשומות המעורבות בתהליך העיבוד. לדוגמא, 10 אחוז פעילות פירושו שבמשך תקופה מסויימת (או בעיבוד מסויים) עודכנו או נקראו 10 אחוז מהרשומות בקובץ. ככל שגדל אחוז הפעילות כן כדאי להשתמש בקובץ סידרתי, ולהיפך — אחוז פעילות נמוך עושה את שיטות הארגון הטבלתי או הישיר/אקראי כדאיות יותר. אם יש לעדכן 50 רשומות בקובץ של 10,000 רשומות, השיטה הסידרתית לא תהיה יעילה מכיוון שהיא תחייב סריקה של כל 10,000 הרשומות.

ב. **פעילות** — פעילות הקובץ מתייחסת לתכיפות התוספות והביטולים המבוצעים בקובץ. יש לזכור שהוספה וביטול של רשומות בקובץ סידרתי מחייבים יצירת קובץ חדש, בעוד שבקובץ טבלתי או ישיר/אקראי ניתן לטפל בתוספות וביטולים ללא העתקת הקובץ.

ג. **גודל** — גודל הקובץ הנו גורם חשוב בבחירת שיטת הארגון. קבצים סידרתיים יכולים להיות בכל גודל שהוא. קבצים טבלתיים וישירים/אקראיים מוגבלים לגודל מתקן האחסון, ומהירות הגישה לרשומות בהם מושפעת בצורה ישירה מגודלם.

ד. **זמן תגובה** — היתרון החשוב במערכת מחשב הוא האפשרות לגישה ישירה ומהירה לנתונים, בעיקר בעבודה בשיטת העיבוד הישיר (on-line). ביישומים

כאלה הכרחי לעבוד עם קבצים המאפשרים גישה ישירה לנתון, ואפשרות זו קיימת בקבצים טבלתיים וישירים/אקראיים בלבד.

ה. מחיר האחסון — קבצים סידרתיים ניתנים לאחסון על גבי מתקני אחסון זולים — סרט מגנטי, כרטיסים מנוקבים, וסרט נייר מנוקב, בעוד שקבצים טבלתיים וישירים/אקראיים דורשים מתקני אחסון יקרים יותר — תקליט מגנטי או תוף מגנטי.

II. חלק שני

תיכנות

1.

שפות תיכנות

מבוא

מחשב סיפרתי הוא מכונה אשר מזינים אותה בנתונים ובהוראות כיצד לטפל בנתונים כדי לקבל את התוצאות הרצויות. קבוצת ההוראות נקרא תוכנית (program) והיא נכתבת על ידי תוכניתן (programmer). יש שהנתונים הם חלק מהתוכנית עצמה ויש שהם מוזנים למחשב נפרד מהתוכנית.
כוונת פרק זה אינה להסביר כיצד עובד המחשב, אלא כיצד לגרום לכך שהמחשב יעשה את רצון האדם, זאת אומרת כיצד להכין תוכנית מחשב.

שפת מכונה (Machine Language) *

שפת מחשב היא שפה המאפשרת להורות למחשב כיצד לפעול על הנתונים המוזנים אליו. עקב טבעם האלקטרוני של מחשבים ניתן להגדירם כמכונות אשר יכולות לייצג שני מצבים בלבד (דוגמת מנורה). שני המצבים הם:

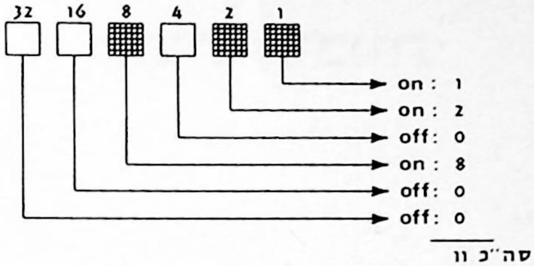
ON   או   OFF

מצבים אלו יכולים להיות מתוארים על ידי הספרות 0 ו־1 (שני הסמלים של שיטת המספרים הבינאריים). כל יחידה כזאת נקראת, כזכור סיבית. כמה יחידות כאלו מאפשרות לייצר מספרים בעזרת צירופים שונים של **אחדים ואפסים**. לדוגמא,

---

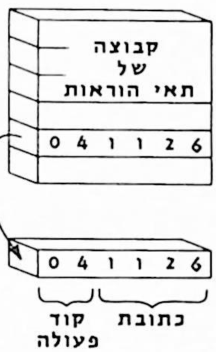
* הערה: כדי לקצר בכתיבה, נסמן להלן את הצירופים הבינריים בערכם העשרוני, אך יש לזכור שההצגה הפנימית במחשב הנה תמיד בשיטה הבינרית.

נניח שש יחידות כאלו קובצו לתא אחד, כך שאם נתחיל מהקצה הימני, הערך שכל יחידה מייצגת הוא 1, 2, 4, 8, 16 .... וכך הלאה (תרשים 2.1).



תרשים 2.1:

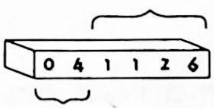
צירופים שונים של הסיביות יכולים לייצג מספרים או תווים אלפאביתיים.
נניח שאחד התאים האלה מכיל ״הוראת״ מחשב. זהו צירוף מסוים של ON ו־OFF (ספרות 0 ו־1), אשר גורם לכך שהמעגלים הפנימיים של המחשב יבצעו פעולה אלקטרונית פשוטה. ההוראה עצמה יכולה גם היא להתחלק לחלקים, כמתואר להלן:



קוד 04 (המיוצר על ידי צירוף של סיביות)
הנו הוראה למחשב לבצע פעולה מסוימת על
תוכן התא בזיכרון שמספרו מופיע בהמשך ההוראה.

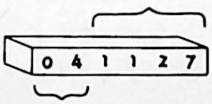
נמשיך ונפתח את הדוגמא ונניח שתי הוראות עוקבות, אשר כל אחת מהן מכילה את קוד ההוראה 04, הגורמת למחשב להדפיס את תוכן התא אשר כתובתו מצוינת בהוראה.

הוראה 1:



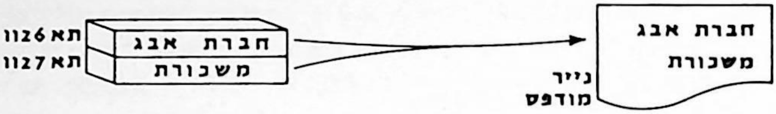
״04״ הנה הוראה להדפסת תוכן תא 1126
״1126״ היא כתובת של תא בזיכרון המכיל בתוכו ״חברת אבג״.

הוראה 2:



״04״ הנה הוראה להדפסת תוכן תא 1127
״1127״ היא כתובת של תא המכיל בתוכו את התוכן ״משכורת״.

כאשר שתי ההוראות הנ"ל מתבצעות ברצף תהיה התוצאה כדלהלן:



הדוגמא לעיל מראה בצורה פשוטה את שפת המחשב הבסיסית הנקראת שפת מכונה.

דוגמא לשפת מכונה

כדי להמחיש את צורת פעולתה של שפת מכונה, נתאר לעצמנו מחשב מדומה אשר גודל המלה שלו מאפשר שימוש ב־9 סיביות. מלה זאת קטנה בהרבה מהמקובל במחשבים רגילים, אולם הדוגמא תאפשר לנו להמחיש את השימוש בשפת מכונה ביתר קלות. כזכור מהפרק הדן באופן פעולת המחשב, יהידת העיבוד המרכזית מורכבת, בין השאר, מיחידה אריתמטית־לוגית, והיא המבצעת את ההוראות הניתנות למחשב. היחידה האריתמטית מכילה אוגר (accumulator), אשר חלק מההוראות בשפת מכונה מתייחס אליו. כל הוראה בשפת המכונה של המחשב המדומה בנויה מ־9 סיביות; שלוש הסיביות הראשונות מגדירות את סוג ההוראה, ושש הסיביות האחרות מתייחסות לכתובת בזיכרון.

× × × × × ×	× × ×
כתובת	סוג ההוראה

כאשר יש אפשרות להשתמש ב־3 סיביות להגדרת סוג ההוראה, ניתן להגדיר שמונה (2^3) סוגי הוראות. הרשימה להלן מסבירה את שמונה סוגי ההוראות. הסימן × מייצג את הסיפרה 0 או 1.

פירוש ההוראה	כתובת	סוג ההוראה
עצור. (בהוראה זאת אין חשיבות לכתובת).	× × × × × ×	000
העתק את המספר מכתובת × × × × × × לתוך האוגר.	× × × × × ×	001
חבר את המספר הנמצא בכתובת × × × × × × למספר הנמצא באוגר. התוצאה תהיה באוגר.	× × × × × ×	010
אחסן את המספר הנמצא באוגר בכתובת × × × × × ×.	× × × × × ×	011
נקב כרטיס אשר יכיל את המספר הנמצא בכתובת × × × × × ×.	× × × × × ×	100

הסתעף להוראה הנמצאת בכתובת ××××××.	××××××	101
אם המספר הנמצא באוגר הוא שלילי — בצע את ההוראה הנמצאת בכתובת ××××××, אם איננו שלילי — המשך בביצוע התוכנית בהוראה העוקבת.	××××××	110
חסר את המספר הנמצא בכתובת ×××××× מהמספר הנמצא באוגר. התוצאה תהיה באוגר.	××××××	111

דוגמא לתוכנית בשפת מחשב

התוכנית מאוחסנת בזיכרון המחשב, ומקומה של כל הוראה בזיכרון (הכתובת) מצויינת משמאלה.

הסבר	כתובת	סוג ההוראה	מקום בזיכרון	(כתובת)
העתק את המספר בכתובת 16 לאוגר.	010000	001	000001	(1)
חבר את המספר הנמצא בכתובת 19 למספר הנמצא באוגר; התוצאה תהיה באוגר.	010011	010	000010	(2)
אחסן את המספר הנמצא באוגר בכתובת 16.	010000	011	000011	(3)
העתק את המספר הנמצא בכתובת 17 לאוגר.	010001	001	000100	(4)
חבר את המספר הנמצא בכתובת 18 למספר הנמצא באוגר; התוצאה תהיה באוגר.	010010	010	000101	(5)
אחסן את המספר הנמצא באוגר בכתובת 17.	010001	011	000110	(6)
חסר את המספר הנמצא בכתובת 15 מהמספר הנמצא באוגר; התוצאה תהיה באוגר.	001111	111	000111	(7)
אם המספר הנמצא באוגר הוא שלילי — עבור להוראה הנמצאת בכתובת 11; אחרת המשך להוראה הבאה (בכתובת 9).	001011	110	001000	(8)
נקב את המספר הנמצא בכתובת 16 על גבי כרטיס.	010000	100	001001	(9)
עצור.	000000	000	001010	(10)

העתק את המספר הנמצא בכתובת 2 לאוגר.	000010	001	001011	(11)
חבר את המספר הנמצא בכתובת 18 למספר הנמצא באוגר ; התוצאה תהיה באוגר.	010010	010	001100	(12)
אחסן את המספר הנמצא באוגר בכתובת 2.	000010	011	001101	(13)
עבור להוראה הנמצאת בכתובת 1.	000001	101	001110	(14)
נתון, כתובת 15.	100000	000	001111	(15)
נתון, כתובת 16.	000000	000	010000	(16)
נתון, כתובת 17.	000000	000	010001	(17)
נתון, כתובת 18.	000001	000	010010	(18)
נתון, כתובת 19.	001111	011	010011	(19)
נתון, כתובת 20.	101001	111	010100	(20)
נתון, כתובת 21.	000011	011	010101	(21)
נתון, כתובת 22.	001110	101	010110	(22)
.			.	
.			.	
.			.	
נתון, כתובת 50.	001101	001	110010	(50)

הערה: בכתובות שבהן מאוחסנים נתונים, כל תשע הסיביות כיחידה אחת מהוות את הנתון.

יש לשים לב שהתוכנית נמצאת ב־14 הכתובות הראשונות בזיכרון ואילו בשאר הכתובות מאוחסנים נתונים. ניתן להשתמש במספרים הנמצאים ב־14 הכתובות הראשונות כנתונים במידה ויש צורך בכך. יש לשים לב שרק ההוראות והנתונים נמצאים בזכרון המחשב, בעוד שהכתובות מובאות לעיל רק כדי להקל על הבנת התוכנית.

ביצוע שלוש ההוראות הראשונות גורם לכך שהמספרים הנמצאים בכתובת 16 ו־19 מחוברים, והתוצאה מאוחסנת בכתובת 19. ביצוע ההוראות הנמצאות בכתובות 4 עד 6 גורם לכך שהמחשב סופר מ־0 ל־1, והתוצאה מאוחסנת בכתובת 17. ביצוע הוראה הנמצאת בכתובת 7 גורם לכך שהתוצאה המאוחסנת באוגר היא 31—. עקב כך, שההוראה הנמצאת בכתבת 8 גורמת להסתעפות להוראה שכתובתה 11. הוראות הנמצאות בכתובות 11 — 13 משנות את ההוראה בכתובת 2 ; בתחילה הוראה זאת היתה ״חבר את המספר בכתובת 19 ...״ וכעת היא ״חבר את המספר בכתובת 20 ...״. התוכנית תמשיך לבצע את ההוראות עד שתגיע לביצוע ההוראה הנמצאת בכתובת 10, ואז יופסק המשך ביצוע התוכנית.

תוכנית זאת מבצעת למעשה חיבור של סידרת מספרים, המאוחסנים בכתובת 19 ועד כתובת 50, ועד בכלל.

כפי שתואר, בפרק 2—I המחשב יכול לבצע מספר פעולות בסיסיות, לדוגמא: לחבר שני מספרים, לקבוע באם מספר מסוים הוא חיובי, שלילי או אפס; וכך הלאה. בשפת המכונה, הוראות אלה בנויות מצירופים של מספרים אפס ואחד, אשר המחשב "מבין", אולם רק אדם בעל ניסיון רב יכול לפענח בקלות את ההוראות הללו. למעשה, האדם צריך לתרגם כל הוראה בשפת מכונה להוראה בשפה המובנת לו. שפת המכונה שונה ממחשב למחשב — לכל מחשב שפת מכונה המיוחדת לו.

השימוש בשפת מכונה מסורבל ומעמיד בפני התוכניתן שתי בעיות חמורות:

א. כתיבת תוכנית בשפת מכונה היא פעולה מסורבלת מכיוון שיש צורך לפרט את כל הפעולות, אף הבסיסיות ביותר, שעל המחשב לבצע.

ב. בעת כתיבת התוכנית יש לזכור בכל עת מהי תכולת כל תא בזיכרון ומהי כתובתה של כל הוראה. בעיה זו חמורה במיוחד במחשבים גדולים בעלי עשרות אלפי תאי זיכרון.

שפה סימבולית (Symbolic Language)

השפה הסימבולית פותחה משפת המכונה, והיא נוחה יותר לשימוש על־ידי אדם. בשפה זו השימוש הוא באותיות, להבדיל מהשימוש בספרות בינריות בשפת המכונה. להלן רשימת ההוראות בשפה סימבולית של המחשב המדומה, במקביל להוראות בשפת המכונה:

פירוש ההוראה (פירוש מפורט ראה בעמוד 79)			ההוראה בשפה סימבולית	ההוראה בשפת המכונה
עצור	—	SToP	STP	000
העתק	—	COPy	COP	001
חבר	—	ADD	ADD	010
אחסן	—	STOre	STO	011
נקב	—	PUNch	PUN	100
הסתעף	—	JUMp	JUM	101
הסתעף בתנאי	—	Jump IF	JIF	110
חסר	—	SUBtract	SUB	111

ברור שלאדם קל בהרבה לזכור, למשל, את משמעות ההוראה STO או JUM מאשר לזכור את משמעות הצירוף 011 או 101, מאחר והאותיות מזכירות את ההוראה בשפה האנגלית. בשימוש בשפה סימבולית, המחשב הוא המתרגם כל

הוראה הכתובה בשפה סימבולית להוראה הכתובה בשפת מכונה ה"מובנת" לו. המחשב משתמש בטבלה דומה ובתוכנית מתרגמת (הנקראת ASSEMBLER) על מנת לתרגם את ההוראות מהשפה הסימבולית לשפת מכונה לפני ביצוע התוכנית. לכל מחשב שפה סימבולית המיוחדת לו. תוכנית בשפת מכונה שהוצגה קודם לכן, תיראה בשפה סימבולית בצורה הניתנת להלן:

הסבר	כתובת	הוראה
העתק את המספר בכתובת SUM לאוגר	(SUM)	(C) COP
חבר את המספר בכתובת Xn למספר באוגר	(Xn)	(B) ADD
אחסן את המספר באוגר בכתובת SUM	(SUM)	STO
העתק את המספר בכתובת COUNT לאוגר	(COUNT)	COP
חבר את המספר בכתובת ONE למספר באוגר	(ONE)	ADD
אחסן את המספר באוגר בכתובת COUNT	(COUNT)	STO
חסר את המספר בכתובת NUMB מהמספר באוגר ואחסן את התוצאה באוגר	(NUMB)	SUB
עבור להוראה A אם המספר באוגר הוא שלילי; אחרת המשך להוראה הבאה	(A)	JIF
נקב את המספר הנמצא בכתובת SUM	(SUM)	PUN
עצור		STP
העתק את המספר B לאוגר	(B)	(A) COP
חבר את המספר בכתובת ONE לאוגר	(ONE)	ADD
אחסן את המספר באוגר בכתובת B	(B)	STO
עבור להוראה C	(C)	JUM
נתונים (בכתובת NUMB נמצא המספר 32)	32	(NUMB) DTA
נתונים (בכתובת SUM נמצא המספר 0)	0	(SUM) DTA
נתונים (בכתובת COUNT נמצא 0)	0	(COUNT) ADT
נתונים (בכתובת ONE נמצא 1)	1	(ONE) DTA

כפי שהוצג לעיל בנויה כל הוראה משתי מלים (פרט להוראת הסיום). המלה הראשונה היא הוראת הביצוע, והמלה השנייה מגדירה כתובת מסוימת בזיכרון. Xn מייצג כתובת בזיכרון אשר בה מאוחסן נתון n באופן זמני (כל אחד מ־50 המספרים אותם מסכמת התוכנית). הוראת DTA מודיעה למחשב שאחריה מופיע נתון; הוראה זו איננה גורמת לביצוע כל פעולה על־ידי המחשב. הסימונים בסוגריים הם שמות המייצגים כתובת בזיכרון; שמות אלה הם שרירותיים ונקבעים על־ידי התוכניתן. לדוגמא (C) JUM יגרום להסתעפות להוראה (C) COP (SUM).

שפות עיליות

בעבר נאלצו תוכניתנים ללמוד את שפות המכונה או השפות הסימבוליות של המחשבים השונים שבהם השתמשו. כדי ליצור אחידות בשפות התכנות של המחשבים השונים הומצאו שפות כלליות המתאימות למחשבים מטיפוסים שונים, ואשר קל לאדם ללמוד אותן ולכתוב באמצעותן תוכנית מחשב. שפות אלה נקראות שפות תכנות עיליות (high level programming languages).

בין שפות התכנות העיליות הנפוצות נכללות:

1. COBOL (COmmon Business Oriented Language) — שפה אשר תוכננה במיוחד לכתיבת תוכניות בשטח עיבוד הנתונים המינהלי.
2. FORTRAN (FOrmula TRANslator) — שפה המשמשת לכתיבת תוכניות ליישומים מדעיים, אולם מאפשרת שימוש בשטחים אחרים.
3. ALGOL (ALGOrithmic Language) — שפה המתחרה עם FORTRAN. השימוש בה רב יותר באירופה מאשר בארה"ב.
4. PL/1 (Programming Language One) — שפה זו מבוססת על שילוב של COBOL ו־FORTRAN.
5. BASIC (Beginners All־purpose Symbolic Instruction Code) — שפת־בת של FORTRAN, אשר פותחה בעיקר עבור משתמשים במערכות שיתוף־זמנים (time sharing). שפה זו היא ללא ספק הקלה ביותר ללימוד.

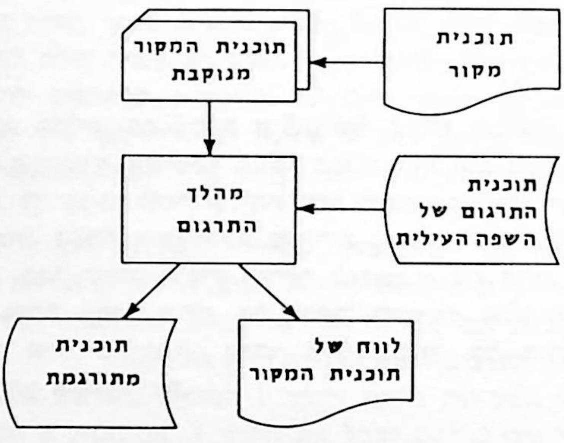
על מנת להמחיש את ההבדל בין שפת תכנות עילית לשפות מכונה ושפות סימבוליות ניתנה להלן דוגמא של תוכנית בשפת BASIC המבצעת את אותו עיבוד שראינו קודם לכן בשפת מכונה ובשפה סימבולית. בדוגמא ניתן לראות שהתוכנית איננה מטפלת בכתובות בזיכרון אלא במשתנים המייצגים כתובות אלו.

מספר שורה	הוראה	הסבר
10	LET I = 0	הצב אפס למשתנה I.
20	LET S = 0	הצב אפס למשתנה S.
30	READ A	קרא נתון מהוראה DATA והצב אותו למשתנה A.
40	LET S = S+A	חבר את ערך משתנה A לערך משתנה S ואת התוצאה הצב במשתנה S.
50	LET I = I+1	חבר אחד לערך משתנה I ואת התוצאה הצב במשתנה I.
60	IF I <= 32 THEN 30	אם ערך המשתנה I אינו גדול מ־32, הסתעף להוראה 30. אחרת המשך להוראה הבאה.
70	PRINT S	הדפס את ערך המשתנה S.
80	DATA 16, 0, 0, 1, 207, 489, 334, ...77	רשימת נתונים.
90	END	עצור

תרגום התוכנית

שפות התכנות העיליות משמשות בסידרת הוראות הדומות מאוד להוראות בשפה האנגלית, ולכן קל לתוכניתן לכתוב תוכניות בשפות אלה. תרגום התוכניות הכתובות בשפה עילית לתוכניות בשפת מכונה נעשה באמצעות תוכניות תרגום מיוחדות (compilers). המתכנת כותב את תוכניתו בשפה עילית (תוכנית מקור), המחשב, בעזרת תוכנית התרגום מתרגם את תוכנית המקור לשפת מכונה, ומבצע את התוכנית לפי ההוראה בשפת המכונה. בתרגום התוכנית לשפת מכונה מתורגמת כל הוראה הכתובה בשפה העילית להוראה או סידרת הוראות בשפת המכונה, ולכל הוראה או משתנה נקבע מקומו בזיכרון וכתובתו.

בכתיבת התוכניות בשפה העילית יש להקפיד על כתיבה מדויקת של ההוראות בהתאם לכללי השפה העילית, אחרת לא תוכל התוכנית המתרגמת להפוך את ההוראות לשפת מכונה. תוכנית התרגום מתרגמת בצורה טכנית ואינה מסוגלת להבין את כוונת המתכנת כאשר היא נתקלת במשפט לא שלם, לא ברור, או שגוי, ומשום כך אינה מסוגלת לתקן שגיאות. תהליך הכנת התוכנית והתרגום (הנקרא קומפילציה — compilation) מתואר בתרשים 2.2.



תרשים 2.2 : תהליך הכנת התוכנית והתרגום

2.

שפת התכנות Basic*

BASIC — Beginner's All-purpose Symbolic Instruction Code

ככבל שפת תכנות עילית, גם לשפת ה־BASIC כללים קבועים של סדר וצורה, על פיהם נכתבות ההוראות. השימוש בשפת תכנות זאת הוא בעיקר בעזרת מסופים הקשורים למחשב, כך שניתן להזין את התוכנית למחשב ולבצעה מיד.
תוכנית (program) בשפה זו מורכבת **משורות** (lines), ולכל שורה **מספר שורה** (line number). בכל שורה תופיע **הוראה** (statement) אחת. ביצוע ההוראות על־ידי המחשב נעשה בסדר עולה של מספרי השורות, אלא אם קיימת הוראה המעבירה למקום אחר בתוכנית. בדרך כלל תכלול כל הוראה **מלת מפתח** (key־word) המזהה את ההוראה.

דוגמא של תוכנית מחשב ב־BASIC

תוכנית ב־BASIC הפותרת שתי משוואות לינאריות: ax + by = c ו־dx + ey = f ניתנת להלן. התוכנית מקבלת את הערכים של e,d,c,b,a ו־f, ופותרת עבור הערכים של x ו־y.

---

* שפת ה־BASIC המתוארת בספר זה מתאימה למחשב CDC 6600 הנמצא באוניברסיטת תל־אביב. יכולים להיות שינויים קלים בכללים או בסימונים בשפות ה־BASIC במחשבים אחרים.

אם bd — ae אינו שווה לאפס, הבעיה ניתנת לפתרון בצורה הבאה:

$$x = \frac{ce - bf}{ae - bd} \quad \text{ו} \quad y = \frac{af - cd}{ae - bd}$$

אם bd — ae שווה אפס, אז אין פתרון יחיד לבעיה (קיימים אינסוף פתרונות).

```
100  READ  A,B,D,E,C,F
110  LET  G = A*E — B*D
120  IF  G = 0 THEN 170
130  LET  X = (C*E — B*F)/G
140  LET  Y = (A*F — C*D)/G
150  PRINT X,Y
160  GO TO 999
170  PRINT "NO UNIQUE SOLUTION"
180  DATA 1,2,4,2,—7,5
999  END
```

ניתן לראות כי 1) הוראות התוכנית כתובות רק באותיות גדולות (capital letters). 2) כל שורה (הוראה) מתחילה עם מספר, הנקרא מספר שורה או מספר הוראה. לפני ביצוע התוכנית, מסדר המחשב את ההוראות בסדר עולה לפי מספרי השורה ומאחסן אותן בזיכרון. מספרי השורות משמשים ככתובות להוראות בתוכנית. ביצוע התוכנית מתחיל מההוראה הראשונה וממשיך לפי סדר ההוראות ובהתאם להוראות ההסתעפות בתוכנית. 3) בכל שורה מלת מפתח (key-word), המגדירה את סוג ההוראה.

נעבור על התוכנית צעד אחר צעד. השורה הראשונה (שורה מספר 100) היא הוראת READ. הוראה זו מחייבת שתהיה בתוכנית לפחות הוראה אחת של DATA או יותר. כאשר מבצע המחשב את ההוראה הזאת, הוא יעביר למשתנים המופיעים בהוראה READ את הערכים המופיעים בהוראת DATA.

בתוכנית זו, יקבל המשתנה A את הערך 1 המופיע בהוראת DATA בשורה 180, המשתנה B את הערך 2, ו־D יקבל את הערך 4. המשתנה E יקבל את הערך 2, C יקבל את הערך 7 — ו־F את הערך 5.

לאחר מכן ממשיך ביצוע התוכנית לשורה 110, הוראת LET. מטרת הוראה זו לחשב את הערך של AE — BD ולהציבו במשתנה G (הסימן * משמעותו כפל). הערך של G בדוגמא שלנו הוא 6—.

יודעים אנו שכאשר G שווה לאפס, אין פתרון אחד לקבוצת המשוואות, ולכן מבצעת התוכנית בדיקה בשורה 120 כדי לוודא אם G שווה לאפס. אם התוכנית מוצאת כי התשובה היא "כן", מסתעפת התוכנית לשורה 170. שורה 170 תגרום להדפסת ההודעה הבאה:

NO UNIQUE SOLUTION

משורה 170 תמשיך התוכנית להוראה הבאה, אולם שורה 180 היא הוראת DATA, שאינה הוראת ביצוע, לכן תמשיך התוכנית לשורה 999 — הוראת END — אשר גורמת להפסקת ביצוע התוכנית.
אם התשובה לשאלה בשורה 120 "האם G שווה לאפס" היא "לא", התוכנית תמשיך לשורה הבאה, כלומר לשורה 130.
התוכנית מוכנה עכשיו לפתור את צמד המשוואות

$$4x + 2y = 5 \qquad x + 2y = -7$$

שורות 130 ו־140 מורות לתוכנית לחשב את הערכים של x ו־y על פי המשוואות המוגדרות בשורות אלה. יש לשים לב כי השימוש בסוגריים הכרחי כדי ש־CE—BF יחולק ב־G. ללא הסוגריים רק BF היה מחולק ב־G והתוצאה היתה

$$x = CF - \frac{BF}{G}$$

שורות 130 ו־140 גורמות לביצוע החישוב, ומציבות את התוצאות 4 ו־5.5— ל־x ו־y בהתאמה.
שורה 150 תגרום להדפסת הערכים של x ו־y. לאחר מכן ממשיך ביצוע התוכנית לשורה 160, אשר מורה למחשב להסתעף לשורה 999, הגורמת להפסקת ביצוע התוכנית.

מושגים והגדרות

תו (character) הוא המרכיב הלוגי הקטן ביותר הקיים בשפה העילית. כל שורה בתוכנית מורכבת מתווים. התווים מבוטאים במחשב באמצעות **צופן בינרי** (binary code), והמאוחסנים בדרך כלל ב־8 סיביות המהוות **בית** (byte).

ניתן לסווג את התווים לשלוש קבוצות:

א. **תו אלפביתי** — (alphabetic character) A — Z

ב. **תו מספרי** (נומרי) — (numeric character) 0 — 9

התווים הללו יחד מכונים אלפאנומריים (alphanumeric).

ג. תווים מיוחדים — (special characters)

תו הרווח (space), פסיק, נקודה, סימנים אלגבריים וכד׳.

קבוע מספרי (Numerical Constant)

קבוע מספרי מייצג ערכים מספריים שאינם משתנים בזמן ביצוע התוכנית. קבוע מספרי נכתב כסידרת תווים מספריים (אחד או יותר), ויכול להכיל סימן (+ או —) ונקודה עשרונית.

קבוע מספרי **שלם** נכתב ללא נקודה עשרונית 4 23

קבוע מספרי **עשרוני** נכתב עם נקודה עשרונית 7.1 .25 0.5

ניתן לכתוב ב־BASIC קבוע מספרי בצורה מעריכית (exponential).

למשל: 5 — 2.7E— שפירושו 2.7— כפול־10^{-5}, או 0.00027—.

בשפת BASIC מודפסים מספרים המכילים עד 6 ספרות משמעותיות בצורת מספר שלם או עשרוני. אחרת יודפס המספר בצורה מעריכית מנורמלית ומעוגל ל־6 ספרות משמעותיות: למשל, המספר 0.00000187 יודפס בצורה 6— 1.87 E.
הצורה המנורמלת היא כאשר הנקודה העשרונית ממוקמת מימין לסיפרה המשמעותית הראשונה.

משתנה מספרי (Numerical Vairable)
משתנה מספרי מייצג ערך מספרי שיכול להשתנות בזמן ביצוע התוכנית.
משתנה מספרי מיוצג ע״י תו אלפביתי בודד או תו אלפביתי ואחריו תו נומרי, כגון: Z, Q1, A.
ערך המשתנה יכול להשתנות בזמן ביצוע התוכנית. למעשה מייצג המשתנה כתובת בזיכרון שבה מאוחסן מספר.
בשפת BASIC מקבל כל משתנה מספרי, שערכו לא נקבע על־ידי הוראה מפורשת בתוכנית, את הערך ההתחלתי 0, אך לא רצוי להשתמש בעובדה זו בשעת כתיבת תוכנית.

אופרטור אריתמטי (Arithmetical Operator)
אופרטורים אריתמטיים משמשים יחד עם קבועים מספריים ומשתנים מספריים וסימני הסוגריים להצגת ביטויים אריתמטיים. האופרטורים האריתמטיים מבוטאים באמצעות הסמלים הבאים:

הסמל	הפעולה	דוגמא	הסבר
+	חיבור	A + B	חבר B ל־A
—	חיסור	A — B	חסר B מ־A
*	כפל	A * B	כפול B ב־A
/	חילוק	A / B	חלק A ב־B
↑ או **	חזקה	A ↑ 2	העלה A בחזקת 2

ביטוי אריתמטי (Arithmetical Expression)
ביטוי אריתמטי הוא קבוע מספרי בודד, או משתנה מספרי בודד וכן כל צירוף מותר שלהם בעזרת האופרטורים האריתמטיים והסוגריים.

למשל: (A↑2 + B↑2) /(3* X)

סימן המינוס (מינוס אונארי) יכול לבוא לפני כל ביטוי אריתמטי ואף אחרי אופרטור אריתמטי, ללא צורך בסוגריים. הסוגריים המותרים לשימוש הם רק סוגריים עגולים ( ), המשמשים לקביעת סדר ביצוע הפעולות.

המחשב קובע את ערכם של הביטויים האריתמטיים לפי ערך הקבועים והמשתנים המספריים המופיעים בהם ברגע חישוב הביטוי. חישוב הביטוי האריתמטי נעשה משמאל לימין, פרט לאופרטורים בעלי עדיפות גבוהה, המבוצעים לפני אופרטורים בעלי עדיפות נמוכה. סדר העדיפות בין האופרטורים הוא (מהגבוה לנמוך):

1. סוגריים
2. חזקה, מינוס אונארי
3. כפל, חילוק
4. חיבור, חיסור

דוגמאות:

א. 2**2+4/2+3 יתבצע בצורה הבאה: קודם תתבצע החזקה 2^2 שתוצאתה 4. לאחר מכן תתבצע החלוקה 4/4 שתוצאתה 1, ואז יבוצע החיבור 1+3 והתוצאה הסופית תהיה 4.

ב. 2**(4/2)+3 יתבצע כך: קודם יחושב הביטוי בסוגריים 4/2 שתוצאתו 2. לאחר מכן תתבצע החזקה 2^2 שתוצאתה 4, ואז יבוצע החיבור 4+3 והתוצאה הסופית היא 7.

ג. 2**2/(3+4) יתבצע כך: קודם יחושב הביטוי בסוגריים 4+3 שתוצאתו 7, לאחר מכן תתבצע החזקה 2^2, ואז יבוצע החילוק 7/4 והתוצאה הסופית היא 1.25.

ד. 2**(4/2+3) יתבצע כך: קודם יחושב הביטוי בסוגריים: ראשית תבוצע החלוקה 4/2 שתוצאתה 2, לאחר מכן יתבצע החיבור 2+3 שתוצאתו 5, ולבסוף תתבצע החזקה 5^2 והתוצאה הסופית היא 25.

ה. 2**((3+4)/2) יתבצע כך: קודם יחושב הביטוי בסוגריים הפנימיים 4+3 שתוצאתו 7. לאחר מכן יחשב הביטוי בסוגריים החיצוניים 7/2 שתוצאתו 3.5, ואז תתבצע החזקה 3.5^2 והתוצאה הסופית היא 12.25.

הצבה (השמה) (Assignment)

סימן השוויון (=) משמש להשמה של ערך הביטוי הנמצא מימינו למשתנה הנמצא משמאלו. בניגוד לכללי האלגברה, מותר בשפות התכנות לרשום אותו משתנה משני צידי השוויון, מכיוון שההשמה נעשית תמיד מימין לשמאל.

דוגמאות:

A = 1 הצב את הערך 1 למשתנה A.

A = 2 + 7/3 חשב את הביטוי 7/3 + 2 והצב את ערך התוצאה למשתנה A.

A = A * 3 חשב את הביטוי A*3 כאשר הערך של המשתנה A הוא ערכו בזמן ביצוע ההוראה והצב את התוצאה למשתנה A (כלומר, הערך החדש של A יהיה פי שלוש מערכו לפני ביצוע ההוראה).

אופרטור־יחס — (Relational Operator) (בקיצור (R)).

בין מספרים קיימים שבעה אופרטורי־יחס הנכתבים כדלקמן:

הסמל	היחס	דוגמא	הסבר
=	שווה	A = B	A שווה ל־B
<	קטן	A < B	A קטן מ־B
>	גדול	A > B	A גדול מ־B
<>	לא שווה	A<>B	A לא שווה ל־B
>=	לא קטן	A>=B	A גדול או שווה ל־B
<=	לא גדול	A<=B	A קטן או שווה ל־B

ביטוי־יחס — (Relation)

נוצר על־ידי שני ביטויים אריתמטיים שביניהם אופרטור יחס. **צורת ביטוי היחס** היא אפוא:

ביטוי אריתמטי (R) ביטוי אריתמטי

ביטוי היחס משמש כ״תנאי״ שיש לו ערך **אמת** או **שקר** (TRUE או FALSE). למשל: 4<8 יקבל את הערך **אמת** ואילו 8<4 יקבל את הערך **שקר**.

תיאור הוראות BASIC

בהוראה יכולים להיכלל לכל היותר 72 תווים. כל הוראה ב־BASIC נכתבת כשורה אחת. לכל הוראה מספר שורה, שיכול להיות בין 1 ל־99999. לפני ביצוע התוכנית מסודרות ההוראות בסדר עולה לפי מספרי השורות. כדי לתאר באופן כללי הוראות BASIC ניעזר בסימון הבא:

< LN > או <LN1>, <LN2>, וכו׳ יסמן מספר שורה (מ־1 עד 99999)
< C > או <C1> וכו׳ יסמן **קבוע מספרי**
< V > או <V1> וכו׳ יסמן **משתנה מספרי**
<EXP> או <EXP1> וכו׳ יסמן **ביטוי אריתמטי** .

READ הוראת הקלט הגורמת לקריאת נתונים. לאחר מלת המפתח תבוא **רשימה** של **משתנים** המופרדים על־ידי תווי פסיק (,). (אסור לשים פסיק בראש הרשימה או בסופה!). קריאת המספרים נעשית מתוך רשימת נתונים המוגדרת על־ידי הוראות DATA (ראה להלן), והמשתנים מקבלים את הערכים הנקראים מרשימת הנתונים לפי סדר הופעתם.

מבנה ההוראה:

<LN> READ <V1>, <V2>, ...

דוגמא:

150 READ X, Y, Z, X1, Y2, Q9

DATA אינה הוראת ביצוע אלא הכרזה (declaration). שורות ה־DATA מופיעות בתוכנית אך אינן משתתפות בביצוע התוכנית. הכרזה זו מכילה רשימת נתונים לשימוש הוראת הקלט READ שבתוכנית. בהמשך השורה מופיעה רשימת קבועים המופרדים בפסיקים. קובץ הנתונים כולו נוצר מכל שורות ה־DATA, לפי סדר עולה של מספריהן.

מבנה השורה:

<LN> DATA <C1>, <C2>, ...

דוגמא:

330 DATA 4,2,1.7,6.734E—3,—174.1

בשעת ביצוע כל הוראת קריאה (READ), מתקדם בתוך קובץ הנתונים מחוון (pointer) המורה תמיד על הנתון הבא המועמד לקריאה. אם נעשה ניסיון לקרוא לאחר שהמחוון הגיע לסוף קובץ הנתונים תודפס על־ידי התוכנית הודעת שגיאה:

OUT OF DATA

או:

NOT ENOUGH DATA

ולאחר הדפסת הודעת השגיאה יופסק ביצוע התוכנית.

NODATA הוראה זו בודקת אם המחוון המורה על הנתון הבא נמצא מעבר לנתון האחרון בקובץ. כדי לבדוק זאת, אפשר לכתוב את ההוראה NODATA לפני ההוראה READ, כך שהתוכנית תבדוק אם נקראו כל הנתונים בקובץ לפני שתנסה לקרוא נתון נוסף. אם כל הנתונים מהקובץ נקראו, התוכנית תסתעף להוראה שמספרה רשום בהוראה NODATA. אם נותרו נתונים נוספים בקובץ (שעדיין לא נקראו) תמשיך התוכנית להוראה הבאה אחרי ההוראה NODATA.

דוגמא: 290 NODATA 210

בדוגמא זו, אם כל הנתונים בקובץ נקראו, התוכנית תסתעף להוראה 210; אחרת — התוכנית תמשיך להוראה הבאה.

RESTORE כשיש צורך לחזור ולקרוא את כל הקובץ מתחילתו, ניתן לשבץ את ההוראה RESTORE בתוכנית. ביצוע הוראה זו מחזיר את המחוון לראש הקובץ.

דוגמא:

340 RESTORE

PRINT הוראת פלט הגורמת להדפסה במסוף. לאחר מלת המפתח תבוא רשימה של משתנים או ביטויים אריתמטיים המופרדים על־ידי פסיקים. כמו כן יכול להופיע אחרי מלת המפתח קבוע אלנומרי המורכב מסידרה כלשהי של תווים הסגורה בין גרשיים (").

מבנה ההוראה:

<LN> PRINT <EXP1> <EXP2> ,....

או

<LN> PRINT "..." <EXP>, <V> ...

להוראת ה־PRINT ישנן כמה מטרות:

א. להדפיס ערכי משתנים וערכים של ביטויים אריתמטיים (תוצאות של חישובים).

ב. להדפיס קבועים אלפנומריים — הודעות הכלולות בתוכנית.

ג. להדפיס שילוב של הנ"ל.

ד. לדלג על שורה במסך או על הנייר המודפס במסוף.

דוגמא למטרה א':

```
100  PRINT  X, X**2
135  PRINT  X,Y,Z,B*B—4*A*C,(A—B)**2
```

השורה הראשונה (שורה מספר 100) תגרום להדפסת הערך של X, ולאחר כמה רווחים באותה השורה את הערך של X^2. השורה השנייה (שורה מספר 135) תגרום להדפסת חמישה מספרים, שלושה שהם הערכים של Y, X ו־Z בהתאמה, ושניים עבור תוצאת הביטויים $B^2 - 4AC$ ו־$(A-B)^2$ (כמובן במידה ולמשתנים המספריים B, A ו־C ניתן ערך מספרי לפני פעולת ה־PRINT).

דוגמא למטרה ב':

```
100 PRINT "NO UNIQUE SOLUTION"
430 PRINT "X VALUE", "SINE", "RESOLUTION"
```

השורה הראשונה (שורה מספר 100) תגרום להדפסת הכתוב בין הגרשיים ("). אותו דבר יתבצע על־ידי ההוראה השנייה (שורה מספר 430) אולם במקרה זה יופרדו שלושת ההודעות על־ידי מספר רווחים (שים לב לסימני הפסיק המפרידים בין שלושת ההודעות).

דוגמא למטרה ג':

```
150 PRINT "THE VALUE OF X IS"; X
315 PRINT "THE SQUARE OF"; X; "IS"; X**2
```

אם נניח, לדוגמא, שהערך של המשתנה המספרי X הוא 25, אז השורה הראשונה (שורה מספר 150) תגרום להדפסת השורה הבאה:

```
THE VALUE OF X IS 25
```

השורה השנייה (שורה מספר 315) תגרום להדפסת השורה הבאה:

```
THE SQUARE OF 25 IS 625
```

דוגמא למטרה ד׳:

```
250 PRINT
```

המחשב יקדם את הדף בשורה אחת כשתתבצע הוראה מספר 250. למעשה, תגרום הוראה זו לרווח כפול (Double Space).

בין הביטויים המופיעים בהוראת ה־PRINT מפרידים שני סוגי תווים, המשמשים בעצמם מעין "הוראה":

א) **תו הפסיק** (,). הוראה זו גורמת לראש ההדפסה במדפסת לעבור ל־"שדה הדפסה" הבא — בכל שורה ישנם 5 שדות הדפסה ברוחב 15 תווים כל אחד.

ב) **תו נקודה־פסיק** (;) — הוראה זו גורמת לראש ההדפסה להישאר במקומו, דבר המאפשר הדפסה של יותר מ־5 שדות בשורה.

תווים אלה יכולים להופיע גם בסוף הרשימה ובראשה. אם בסוף הרשימה לא מופיע תו כזה, מפרש זאת המחשב כהוראה לעבור לתחילת השורה הבאה. אי לכך ההוראה PRINT בעצמה, ללא כל רשימה של משתנים ו/או ביטויים, גורמת ליצירת רווח של שורה אחת בהדפסה.

LET — הוראת ההצבה והחישוב.

מבנה ההוראה:

```
<LN> LET <V> = <EXP>
```

ביצוע הוראה זו גורם לחישוב (אם דרוש) של הביטוי <EXP> והצבת ערכו במשתנה <V>. (הוראה זו אינה הוראת שוויון אלגבראית ואינה אופרטור היחס שווה).

ניתן להשמיט את המלה LET ללא שינוי משמעות ההוראה, (זו מילת המפתח היחידה הניתנת להשמטה).

ההוראה X = 3 תגרום לכך שהמשתנה המספרי X יקבל את ערך הקבוע המספרי 3.

ההוראה X = Y תגרום לכך שערכו של המשתנה Y יוצב למשתנה X.

ההוראה: LET X = X+2 תגרום להגדלת ערך המשתנה המספרי X ב־2.

דוגמאות:

```
100 LET X = X+1
259 LET W7 = (W—X4↑3) * (Z—A) / (A—B) — 17
```

ניתן להציב ערך אחד למשתנים רבים על־ידי הצורה הבאה:

```
<LN> LET <V1> = <V2> = <V3> ...     = <EXP>
```

ערך הביטוי <EXP> יוצב בכל אחד מהמשתנים המופיעים משמאלו. גם במקרה זה מותר להשמיט את המלה LET.

END — כל תוכנית חייבת להסתיים בהוראה זאת ולכן היא חייבת להיות בעלת מספר השורה הגבוה ביותר, כלומר — זו ההוראה האחרונה בתוכנית.

סיום נורמלי של תוכנית חל כאשר מגיעה התוכנית להוראה זו. אם דרוש להפסיק את ביצוע התוכנית לפני שהגיעה לסיומה ניתן להסתעף להוראת ה־END.

דוגמא:

```
999 END
```

STOP — הוראה זו גורמת להפסקת ביצוע התוכנית. ההוראה יכולה להופיע מספר פעמים ובכל מקום בתוכנית, ואינה חייבת להיות ההוראה האחרונה בתוכנית, אך כאשר ביצוע התוכנית מגיע אליה — מופסק הביצוע.

דוגמא:

```
87 STOP
```

תוכנית לדוגמא

תוכנית לחישוב הערך העתידי של סכום כסף C, כעבור N שנים, לפי שער ריבית R.

הערך העתידי בשנה N מחושב לפי הנוסחה $P = C(1+R)^n$.

```
10 READ C,R,N
20 LET P = C*(1+R)**N
30 PRINT "RESULTS FOR C="; C,"R=";R,"N=";N
40 PRINT "FUTURE VALUE IS";P
50 DATA 10,0.2,3
60 END
```

הפלט מהתוכנית יהיה:

```
RESULTS FOR C=10        P=.2        N=3
FUTURE VALUE IS 17.28
```

REM — (קיצור של remark) — שורת הערה.

אם בראש השורה, לאחר מספרה, מופיעה המלה REM תתייחס המערכת אל שורה כזו כאל הערה; השורה מודפסת בתדפיס התוכנית, אך אין התוכנית מתייחסת אליה בזמן הביצוע.

דוגמא:

```
10 REM THIS PROGRAM COMPUTES AN AVERAGE
```

הערה מאפשרת לתוכניתן להכניס הסברים והערות לתוך התוכנית, כדי להקל על כתיבתה ותיקונה.

INPUT — לעתים יש צורך להזין נתונים תוך מהלך ביצוע התוכנית. דבר זה דרוש כאשר התוכנית מיועדת לעיבוד באמצעות מסוף, והנתונים מוזנים לתוכנית דרך המסוף. פעולה זו ניתנת לביצוע על־ידי השימוש בהוראת ה־INPUT, הדומה במידה מסוימת להוראת ה־READ, אולם אינה דורשת את קיום הכרזות ה־DATA.

מבנה ההוראה: לאחר מלת המפתח INPUT מופיעה רשימת משתנים המופרדים בתווי פסיק:

<LN> INPUT <V1>, <V2>, <V3>, . . .

בהגיע התוכנית להוראה זו, היא מדפיסה על המסוף סימן שאלה (?), פעולתה נעצרת, והיא ממתינה לקבלת הנתונים. על מפעיל המסוף להזין את הנתונים על־ידי הקשת רשימה של קבועים במספר המתאים למספר המשתנים המופיעים לאחר המלה INPUT. ערכי הקבועים שהמשתמש מקיש חייבים להיות מופרדים על־ידי תווי פסיק. לאחר הקשת הנתונים על המשתמש ללחוץ על מקש המשלוח שעל גבי המסוף (carriage return) הגורם להעברת הנתונים לתוכנית, ואז מציבה התוכנית את ערכי הקבועים למשתנים המתאימים וממשיכה בעיבוד. במידה והמשתמש מקיש מספר ערכים גדול מדי, יודיע המחשב על כך, והמשתמש יתבקש להזין את הנתונים שוב. אם מספרם קטן מדי, יודיע המחשב גם על כך והמשתמש יתבקש להוסיף את הנתונים החסרים.
לעתים קרובות, הוראת ה־INPUT תוקדם על־ידי הוראת PRINT, אשר תדפיס למשתמש הנחיות המורות לו אלו נתונים התוכנית דורשת.

לדוגמא, אם התוכנית תכלול את ההוראות הבאות:

```
120 PRINT "YOUR VALUES OF X,Y, AND Z ARE"
130 INPUT X,Y,Z
```

המחשב ידפיס:

```
YOUR VALUES OF X,Y, AND Z ARE
?
```

לאחר מכן ימתין המחשב להכנסת שלושת הנתונים על־ידי המשתמש. סימן השאלה הוא כתוצאה מהוראת ה־INPUT.

תוכנית לדוגמא

מבצעת את אותו חישוב כבדוגמא הקודמת, ומדגימה את השימוש בהוראות INPUT, REM.

```
10 REM PROGRAM TO COMPUTE THE FUTURE
20 REM VALUE OF C
30 PRINT "TYPE THE VALUES OF C, R AND N"
```

```
40 INPUT C,R,N
50 LET P = C*(1+R)**N
60 PRINT "RESULTS FOR C="; C, "R="; R, "N="; N
70 PRINT "FUTURE VALUES IS "; F
80 END
```

הפלט יהיה:

```
TYPE THE VALUES OF C, R AND N
? 10,0.2,3
RESULTS FOR C= 10      R= .2      N= 3
FUTURE VALUE IS 17.28
```

GOTO — הוראת הסתעפות (branch).
לעתים רוצה התכניתן שהתוכנית לא תתבצע לפי סדר הופעת ההוראות, אלא תסתעף להוראה הנמצאת במקום אחר בתוכנית. הסתעפות זו ניתנת לביצוע על־ידי שימוש בהוראה GOTO.

צורת ההוראה:

<LN> GOTO <LN1>

משמעות הוראה זו העברת ביצוע התוכנית להוראה בשורה מספר <LN1>.

דוגמא:

```
150 GOTO 75
```

ON-GOTO — הוראת ההסתעפות הרב־כיוונית.
מבנה ההוראה:

<LN> ON <EXP> GOTO <LN1>, <LN2>,<LN3>, . . . , <LNm>

<LNi> הם מספרי שורות בתוכנית (פסיק ביניהם).

כאשר מגיע הביצוע להוראה זו, מחושב הערך של <EXP> ונקטע למספר שלם. מספר שלם זה חייב להיות בין 1 לבין מספר מספרי השורות המופיעים לאחר המלה GOTO (כלומר $m \geqslant$ ערך $\langle EXP \rangle \geqslant 1$).

אם הערך השלם קטן מ־1 או גדול מ־m תופיע הודעת שגיאה כדלקמן:

```
ON STATEMENT OUT OF RANGE
```

ההסתעפות תתבצע לשורה <LNi> כאשר הערך השלם של <EXP> הוא i.

דוגמא:

```
230 ON X + Y GOTO 575, 490, 150
```

הוראה זו תחשב את ערך הביטוי X+Y, ובהתאם לתוצאה תקבע את מספר ההוראה אליה תסתעף התוכנית.

אם התוצאה אינה מספר שלם, תקטע התוכנית את החלק העשרוני של הוצאה, ותתייחס רק לחלק השלם. למשל, אם התוצאה היא 2.7, התוכנית תתייחס אליה כאילו היא 2.

אם ערך הביטוי 1, תסתעף התוכנית להוראה 575.
אם ערך הביטוי 2, תסתעף התוכנית להוראה 490.
אם ערך הביטוי 3, תסתעף התוכנית להוראה 150.
אם ערך הביטוי קטן מ־1 או לא קטן מ־4, תדפיס התוכנית הודעת שגיאה.

IF THEN — הוראת התנאי. הוראה זו מאפשרת להסתעף מהסדר הרגיל של ביצוע הוראות התוכנית, כאשר יחס לוגי מסוים מתקיים, אחרת יימשך ביצוע התוכנית בצורה הרגילה, זאת אומרת, ההוראה הבאה תבוצע.

צורתה הכללית של הוראה זו תהיה:

```
<LN> IF <REL> THEN <LN1>
```

<REL> הוא ביטוי יחס המשמש כתנאי.
<LN1> הוא מספר שורה של הוראה כלשהי בתוכנית (ההוראה <LN1> מכונה "ההוראה המותנית").
משמעות ההוראה היא שההוראה <LN1> תתבצע אם התנאי <REL> הוא אמיתי. אחרת תתבצע ההוראה שבשורה העוקבת.

דוגמאות:

```
340 IF X<=M THEN 630
420 IF G=O THEN 165
```

בשורה 340, אם ערך המשתנה המספרי X קטן או שווה לערך המשתנה המספרי M, תסתעף התוכנית להוראה 630, אחרת תתבצע ההוראה הבאה בתוכנית.
בשורה 420, אם המשתנה המספרי G שווה לאפס, תסתעף התוכנית להוראה 165, אחרת תתבצע ההוראה הבאה בתוכנית.
בעזרת הוראה זו ניתן להסתעף לכל שורה בתוכנית — הנמצאת לפני או אחרי השורה בה נבדק התנאי.

תוכנית לדוגמא

מבצעת את החישוב כבדוגמא הקודמת.
מדגימה את השימוש בהוראות GOTO ו־IF־THEN.
שים לב שהתוכנית חוזרת על החישוב מספר פעמים עבור ערכים שונים של המשתנים ובודקת אם הריבית (R) שווה לאפס.

```
10  REM PROGRAM TO COMPUTE THE FUTURE
20  REM VALUE OF C
30  NODATA 150
40  READ C,R,N
```

```
50  IF R=O THEN 120
60  LET P = C*(1+R)**N
70  PRINT "RESULTS FOR C="; C,"R=";R,"N=";N
80  PRINT "FUTURE VALUS IS "; P
90  DATA 10,0.2,3,10,0.0,3,10,0.3,3
100 PRINT
100 GO TO 30
120 PRINT "INTEREST RATE = 0 FOR C=";C;"N=";N
130 PRINT
140 GO TO 30
150 END
```

הפלט יהיה:

```
RESULTS FOR C= 10        R=  .2        N=  3
FUTURE VALUE IS 17.28
INTEREST RATE = 0 FOR C= 10  N= 3
RESULTS FOR C= 10        R=  .3        N=  3
FUTURE VALUE IS 21.97
```

FOR-TO-STEP — הוראת התחלת לולאה
NEXT — הוראת סיום לולאה

לעתים נדרש בתוכנית לבצע קטע כלשהו של התוכנית מספר רב של פעמים. הוראת הלולאה (Loop), המורכבת משתי הוראות נפרדות, מאפשרת לבצע סידרת הוראות הנמצאת ביניהן כמספר הפעמים הנדרש. הלולאה מתחילה בהוראת התחלת הלולאה, התוכנית מבצעת את ההוראות הנמצאות אחרי הוראה זו, וכאשר מגיעה התוכנית להוראת סיום הלולאה חוזרת התוכנית ומבצעת שוב את ההוראות הנמצאות אחרי הוראת התחלת הלולאה, עד שהוראות אלה מבוצעות כמספר הפעמים הנדרש.

הצורה הכללית של ההוראה היא:

```
<LN> FOR <V> = <EXP1> TO <EXP2> STEP <EXP3>
          .  סידרת
          .
          .
          .  הוראות
<LN> NEXT <V>
```

לכל הוראת התחלת לולאה המופיעה בתוכנית חייבת להיות גם הוראת סיום לולאה. אם חסרה אחת מהן, תודיע התוכנית על שגיאה. הוראת סיום הלולאה חייבת להופיע אחרי הוראת התחלת הלולאה.

<V> חייב להיות משתנה מספרי (לא ביטוי!). הוא נקרא "משתנה הלולאה".

<EXP1>, <EXP2> <EXP3> יכולים להיות ביטויים אריתמטיים כלשהם.

במהלך הביצוע, כאשר התוכנית מגיעה להוראת התחלת הלולאה, מחושבים הערכים של שלושת הביטויים האריתמטיים הנ"ל לפי ערכם המספרי של הקבועים והמש־תנים באותו זמן.

הערך של <EXP1> נקרא "ערך התחלה" של <V>, הערך של <EXP2> נקרא "הערך הסופי" שלו, וערכו של <EXP3> הוא "ערך הצעד" (step value) של <V>. ערך זה יכול להיות מספר חיובי או שלילי (אך לא אפס!).

הערה: אם STEP <EXP3> לא מופיע בהוראת התחלת הלולאה, יהיה ערך הצעד 1+.

שלושת הערכים הללו אינם משתנים בזמן ביצוע הלולאה. הם יכולים להשתנות רק אם נפנה חזרה אל הוראת ההתחלה. אפילו אם נשנה בתוך הלולאה את הערכים של המשתנים המרכיבים את <EXP1>, <EXP2>, ו־<EXP3>, תעבוד הלולאה לפי שלושת הערכים שחושבו ברגע התחלתה. (דבר זה איננו נכון לגבי משתנה הלולאה, אשר אפשר לשנות את ערכו באמצעות ההוראות שבתוך הלולאה).

ערך משתנה הלולאה <V> "נבדק" בסיום כל מחזור של הלולאה. פירוש הדבר שכאשר מגיעים לסוף הלולאה מוסיפה התוכנית לערך משתנה הלולאה את ערך הצעד ובודקת אם כתוצאה מכך "עבר" ערך המשתנה את הערך הסופי. אנו משתמשים כאן במלה "עבר" משום שהצעד יכול להיות חיובי או שלילי. אם ערך הצעד חיובי, פירוש הדבר כי ערך המשתנה גדל מעל הערך הסופי (אם הוא שווה — לא עבר). אך אם ערך הצעד שלילי, פירושו של "עבר" הוא כי ערך המשתנה קטן מהערך הסופי. כלומר, כאשר מסתיים ביצוע הלולאה, יהיה למשתנה הלולאה הערך האחרון שעדיין אינו עובר את הערך הסופי. בסיום ביצוע הלולאה ממשיכה התוכנית להוראה הבאה אחרי הוראת סיום הלולאה.

דוגמאות:

א.

```
456 FOR J = 1 TO 12
  .
  .
  .
  .
480 NEXT J
```

הערך ההתחלתי של משתנה הלולאה הוא 1, גודל הצעד הוא 1, הלולאה תתבצע 12 פעמים, והערך הסופי של משתנה הלולאה יהיה 12.

ב.

```
230 FOR F3 = —3 TO 12 STEP 2
  .
  .
  .
270 NEXT F3
```

הערך ההתחלתי של משתנה הלולאה הוא 3—, גודל הצעד הוא 2, הלולאה תתבצע 8 פעמים, והערך הסופי של משתנה הלולאה יהיה 11.

ג.

```
100 FOR X = 1 TO 10 STEP 2
110 LET X = X*2
  .
  .
  .
150 NEXT X
```

הערך ההתחלתי של משתנה הלולאה הוא 1, גודל הצעד הוא 2, אך משתנה הלולאה משנה את ערכו גם בתוך הלולאה, ולכן הלולאה תתבצע רק 3 פעמים, והערך הסופי של משתנה הלולאה יהיה 10.

ד.

```
130 FOR A  = 0 TO 3 STEP D
  .
  .
  .
180 NEXT A
```

הערך ההתחלתי של משתנה הלולאה הוא 0, גודל הצעד כערך המשתנה המספרי D (שחייב להיות שונה מ־0). מספר הפעמים שהלולאה תתבצע תלוי בערך של D, וכך גם ערכו הסופי של משתנה הלולאה.

ה.

```
340 FOR K2 = (17 + Z*2)/3 TO 3 * Z STEP 1/4
  .
  .
  .
380 NEXT K2
```

גודל הצעד הוא 0.25. הערך ההתחלתי של משתנה הלולאה, מספר הפעמים שהלולאה תתבצע, והערך הסופי של משתנה הלולאה תלויים בערך המשתנה המספרי Z.

ו.

```
150 FOR S = 2 TO —10
•
•
•
170 NEXT S
```

הערך ההתחלתי של משתנה הלולאה הוא 2, גודל הצעד הוא 1, הלולא תתבצע פעם אחת בלבד (מכיוון שהערך הסופי קטן מהערך ההתחלתי כבר בפעם הראשונה), והערך הסופי של משתנה הלולאה יהיה 2.

חוקי השימוש בהוראות הלולא

1. מותר לשנות בתוך הלולאה את ערכו של משתנה הלולאה, אך יש לשים לב לכך כי הדבר עשוי לשנות את מספר הפעמים שהלולאה מתבצעת.
2. מותר לצאת מלולאה, לבצע הוראות שאינן כתובות בתוך הלולאה, ולחזור אליה בתנאי שהדבר הגיוני — כלומר שאפשר להמשיך ולבצע את הלולאה שאליה חוזרים לפי הערך הסופי וערך הצעד שנקבעו בכניסה הראשונה ללואה.

   מובן שמותר לצאת מלולאה ובכלל לא לחזור אליה, במקרה כזה ערך משתנה הלולאה יהיה ערכו ברגע היציאה מהלולאה.
3. בתוך לולאה אחת יכולה להופיע לולאה שנייה בתנאי שהלולאה הפנימית כלולה כולה בתוך הלולאה החיצונית. במקרה כזה בכל מעבר של הלולאה החיצונית תתבצע הלולאה הפנימית במלוא מספר מעבריה. מספר הלולאות הפנימיות היכולות להופיע בתוך לולראה חיצונית הנו לכל היותר 10.

דוגמא:

```
•
•    הוראות
•
30 FOR I = 1 TO 10
•
•    הוראות
•
50 FOR J = —5 TO 30 STEP 2
•
•    הוראות
•
90 NEXT J
•
•    הוראות
•
```

```
110 FOR I3 = 1TO 10+I
•
•        הוראות
•
150 FOR J = I TO I3
•
•        הוראות
•
190 FOR K = 100 TO I STEP —2
•
•        הוראות
•
230 NEXT K
•
•        הוראות
•
270 NEXT J
•
•        הוראות
•
310 NEXT I3
•
•        הוראות
•
340 NEXT I
•
•        הוראות
•
```

יש לשים לב שההתחלה והסיום של כל לולאה פנימית מוכלים שניהם בתוך הלולאה החיצונית. אם תנאי זה אינו מתקיים תיווצר הודעת שגיאה בנוסח של:

FOR WITHOUT NEXT

או

NEXT WITHOUT FOR

לצורת תוכנית זו קוראים LOOP NESTING.

4. אם צריך לפסוח (בתנאים מסוימים) על ביצוע חלק מההוראות בלולאה, יש להקפיד להפנות תמיד את הביצוע אל הוראת הסיום של אותה לולאה. אם יופנה הביצוע לראש הלולאה, תראה בכך התוכנית כוונה להתחיל מחדש ולבצע את הלולאה.

למשל בקטע התוכנית המובא להלן יודפסו המספרים מ־1 עד 10 **פרט** למספר 6.

```
10 FOR I= 1 TO 10
20 IF I = 6 THEN 40
30 PRINT I
40 NEXT I
```

תוכנית לדוגמא

התוכנית מחשבת את הערך העתידי של סכום כסף C, בשער ריבית R, עבור תקופה של שנה, שנתיים, שלוש שנים, ארבע, חמש ושש שנים.

```
10 REM PROGRAM TO COMPUE THE FUTURE
20 REM VALUE OF C
30 REM FOR 1 TO 6 PERIODS
40 READ C,R
50 FOR N = 1 TO 6
60 IF R=0 THEN 130
70 LET P = C*(1+R)**N
80 PRINT "RESULTS FOR C="; C, "R="; R,="N=";N
90 PRINT "FUTURE VALUE IS  ";P
100 DATA 10,0.2
110 PRINT
120 GO TO 150
130 PRINT "INTEREST RATE = 0 FOR C=";C; "N=";N
140 PRINT
150 NEXT N
160 END
```

הפלט יהיה:

```
RESULTS FOR C= 10          R= .2        N= 1
FUTURE VALUE IS 12.
RESULTS FOR C= 10          R= .2        N= 2
FUTURE VALUE IS 14.4
RESULTS FOR C= 10          R= .2        N= 3
FUTURE VALUE IS 17.28
RESULTS FOR C= 10          R= .2        N= 4
FUTURE VALUE IS 20.736
RESULTS FOR C= 10          R= .2        N= 5
FUTURE VALUE IS 24.8832
RESULTS FOR C= 10          R= .2        N= 6
FUTURE VALUE IS 29.8598
```

GOSUB הוראות הסתעפות לשגרה סגורה
RETURN הוראת חזרה משגרה סגורה

לעתים קיימת סידרת הוראות בתוכנית המבצעת תהליך מסוים ומופיעה באותה צורה במקומות שונים בתוכנית. לדוגמא — קטע תוכנית המקבל סידרת ערכים מספריים ומבצע חישוב הממוצע של ערכים אלה יכול להופיע בכמה מקומות בתוכנית, ובכל מקום לבצע את חישוב הממוצע לסידרת מספרים שונה. בכל מקום בו יופיע קטע זה, תופיע למעשה אותה סידרת הוראות. קטע זה נקרא **שגרה** (subroutine).

כאשר קטע כזה כתוב בכמה מקומות בתוכנית הוא נקרא *שגרה פתוחה*. קיימת אפשרות לחסוך בכתיבת הקטע מספר פעמים על־ידי כתיבתו פעם אחת בלבד בתוכנית, ובכל פעם שיש לבצע את קטע ההוראות תסתעף התוכנית למקום בו נמצא הקטע הדרוש, תבצע את ההוראות המופיעות בו, ותחזור להמשך ביצוע התוכנית מהמקום אליו הסתעפה אל אותו קטע. קטע כזה, הכתוב פעם אחת בלבד בתוכנית, והתוכנית מסתעפת כאשר נדרש לבצע אותו, נקרא **שגרה סגורה**. (תרשים 2.3).



תרשים 2.3: שגרה פתוחה ושגרה סגורה

ההוראה GOSUB המופיעה בתוכנית הראשית "קוראת" לשגרה הסגורה, הוראות השגרה מתבצעות, ובסיום קטע ההוראות ממשיכה התוכנית ומבצעת את ההוראה המופיעה אחרי הוראת ה־GOSUB.

ההסתעפות לשגרה סגורה נעשית על־ידי ציון מספר השורה בה מתחילה השגרה. צורת ההוראה היא

<LN> GOSUB <LN1>

דוגמא:

85 GOSUB 1000

ההוראה RETURN חייבת להיות ההוראה האחרונה בשגרה הסגורה. כאשר התוכנית תבצע את ההוראות המופיעות בשגרה ותגיע להוראה זו, תחזור התוכנית מהשגרה ותבצע את ההוראה העוקבת אחרי הוראת ה־GOSUB אשר גרמה להסתעפות לשגרה.

צורת ההוראה היא

<LN> RETURN

דוגמא:

1200 RETURN

כאמור לעיל, ניתן להסתעף לשגרה סגורה מכל מקום בתוכנית, ובסיום ביצוע השגרה ימשיך הביצוע בהוראה המופיעה אחרי ההוראה שקראה לשגרה.
ניתן להסתעף לשגרה סגורה אחת מתוך שגרה סגורה שנייה ואפילו להסתעף לשגרה הסגורה עצמה כאשר הביצוע נמצא כבר בתוכה. יש להקפיד שהתוכנית לא תסתעף לתוך שורות של שגרה סגורה מבלי שנשלחה לשם על־ידי הוראת GOSUB (הדבר יגרום לשגיאה בביצוע והמחשב יודיע את ההודעה הבאה: RETURN WITHOUT GOSUB).

אין להסתעף על־ידי הוראת GOSUB לקטע תוכנית שאינו שגרה סגורה, זאת אומרת שאינו מסתיים על־ידי הוראת RETURN (השגיאה, במקרה כזה, אינה מתגלה על־ידי הודעת שגיאה, אלא על־ידי ביצוע לא תקין של התוכנית!).

DEF FN הוראת הגדרת הפונקציה
FN שם הפונקציה.
אם בתוכנית נעשה שימוש בנוסחאות חישוב ארוכות, המופיעות באותה צורה במקומות רבים בתוכנית, ומבצעות אותו חישוב על ערכים שונים בכל מקום בו הן מופיעות, ניתן להגדיר נוסחאות אלה פעם אחת בלבד ולחסוך את כתיבתן בכל המקומות בהם נעשה שימוש בהן. נוסחה המוגדרת פעם אחת ונעשה בה שימוש במקומות שונים בתוכנית נקראת *פונקציה* (function).

אם, למשל, יש לחשב במקומות שונים בתוכנית את הערך של הנוסחה

$$A^2X + (2X^2AB^3/(3 + 2AX^5 - B^2A/5) - 10)X^3A^2B$$

כך שבכל מקום הערכים של המשתנים המספריים A ו־B קבועים באותו מקום בתוכנית, והערך של המשתנה המספרי X עשוי להשתנות, נוח יותר לכתוב את הנוסחה פעם אחת בלבד בתוכנית, ולאחר מכן להשתמש בה על־ידי סימן מיוחד המזהה אותה. כתיבת הנוסחה פעם אחת בלבד עדיף גם מהסיבה שהסיכוי לשגיאה בכתיבתה קטן. בנוסף לכך, אם נוסחה זו הנה רק חלק מנוסחה ארוכה ומסובכת יותר, קל בהרבה להשתמש בסימון המיוחד לנוסחה זו במקום לכתוב את כולה.
הגדרת הפונקציה וקביעת שמה נעשית על־ידי ההוראה DEF FN, שצורתה:

<LN> DEF FN□ (V) = <EXP>

מילת המפתח DEF פירושה "הגדר" (define), FN□ מסמל את שם הפונקציה המוגדרת, כאשר הסימן □ מייצג אות מאותיות האלפבית (האנגלי), V מייצג משתנה מספרי שהוא המשתנה של הפונקציה (ארגומנט), והנוסחה מופיעה כביטוי

אריתמטי מימין לסימן השוויון. הגדרת הפונקציה מוגבלת להוראה אחת בלבד, ואין אפשרות להמשיך את ההגדרה בשורות אחרות.

דוגמאות:

```
100 DEF FNA(X) = X**2 + 2*X + 5
150 DEF FNA(A) = A*R*P + R**2*A/((3*P + A)
150 DEF FND(A) = A*R*P + R**2 *A/(3*P + A)
```

בדוגמא הראשונה, X הוא משתנה הפונקציה; כאשר יעשה שימוש בפונקציה יוצב הערך של משתנה הפונקציה ויבוצע חישוב הנוסחה. תוצאת החישוב תוצב ל־FNA(X).

בדוגמא השנייה משתנה הפונקציה הוא A, ומשתתפים בביטוי שני משתנים מספריים אחרים — R ו־P. את ערכם של המשתנים R ו־P יש לקבוע בתוכנית לפני השימוש בפונקציה, ואילו ערך משתנה הפונקציה נקבע בזמן השימוש בה. תוצאת החישוב תוצב ל־FND(A).

השימוש בפונקציה נעשה על־ידי כתיבת שם הפונקציה וציון משתנה הפונקציה. שם הפונקציה יכול להופיע כמשתנה מספרי בתוך ביטוי אריתמטי.

דוגמאות:

```
180 LET Y = 3*FNA(X) + 5/FND(A)
310 LET B = FNA(Z)**2
```

בדוגמא הראשונה מוצב ערך משתנה הפונקציה X ומחושב הערך של FNA (כפי שהוגדר בהוראת DEF FNA(X)), לאחר מכן מתבצע הכפל של FNA(X) ב־3, לאחר מכן מוצב ערך משתנה הפונקציה A ומחושב הערך של FND, מתבצעת החלוקה של 5 בערך של FND(A), ולבסוף מתבצע חיבור תוצאות הביניים שהתקבלו, וערך התוצאה הסופית מוצב למשתנה המספרי Y.

בדוגמא השניה, מוצב ערך משתנה הפונקציה Z ומחושב הערך של FNA, לאחר מכן מחושב הערך של FNA בריבוע, ותוצאת החישוב תוצב במשתנה המספרי B. שים לב שבהגדרת הפונקציה FNA משתנה הפונקציה סומן ב־X, ואילו בזמן השימוש בפונקציה משתנה הפונקציה הוא Z: התוכנית מציבה את ערך משתנה הפונקציה בביטוי לצורך החישוב, ולכן מותר להשתמש בכל משתנה מספרי.

כאשר נעשה שימוש בפונקציה כבמשתנה מספרי בתוך ביטוי אריתמטי, עדיפות הפונקציה (ראה עמוד 90) הנה כעדיפות הסוגריים, כלומר — חישוב ערך הפונקציה יתבצע לפני ביצוע האופרטורים האחרים.

בזמן השימוש בפונקציה, משתנה הפונקציה יכול להיות ביטוי אריתמטי כלשהו, והוא יכול לכלול גם פונקציות אחרות (אך לא את הפונקציה עצמה).

דוגמאות:

```
290 LET S = T + 2*FNA(T + U/2) + 1/(FND(U)*T*3)
940 LET K = FND((Q*4*FNA(Q**3—M/(N + Q))) —2*Q)
```

בתוכנית אחת אפשר להגדיר לכל היותר 26 פונקציות שונות, וזאת מהסיבה ששם הפונקציה מורכב משלוש אותיות בלבד, אשר שתיים מהן קבועות — FN, והשלישית חייבת להיות אות מאותיות האלפבית (בו 26 אותיות). קיימת אפשרות לשנות הגדרת פונקציה בזמן ביצוע התוכנית על־ידי כתיבת הוראת DEF FN לפונקציה שכבר הוגדרה. כאשר יעשה שימוש בפונקציה, הגדרתה היא בהתאם להגדרה האחרונה עליה עברה התוכנית.

קיימות מספר נוסחאות אשר השימוש בהן נפוץ. כמה נוסחאות כאלה הוגדרו בשפת ה־BASIC, כך שהתוכניתן איננו צריך להגדיר אותן כדי להשתמש בהן בתוכניתו. נוסחאות אלה נקראות **פונקציות פנימיות** (built-in functions), ושמן מורכב משלוש אותיות. כדי להשתמש בהן יש לכתוב את שם הפונקציה ואת משתנה הפונקציה. מותר להשתמש בפונקציות אלה כבמשתנים אריתמטיים.
משתנה הפונקציה יכול להיות ביטוי אריתמטי כלשהו, והוא יכול להכיל גם פונקציות אחרות. להלן נסמן ב־X את משתנה הפונקציה.

פונקצית השורש הריבועי SQR(X)
מקבלת את ערך השורש הריבועי של משתנה הפונקציה X אם ערכו חיובי או אפס. אם ערכו של המשתנה X שלילי, תודפס הודעת שגיאה וביצוע התוכנית יופסק.
דוגמא לשימוש:

```
320 LET S2 = SQR(FNA(M))
```

התוכנית תחשב את ערך הפונקציה FNA בהתאם לערכו של M; אם ערך הפונקציה FNA לא שלילי — תחשב התוכנית את השורש הריבועי של ערך הפונקציה, ותציב את הערך שהתקבל למשתנה המספרי S2, אם ערך הפונקציה FNA שלילי — תודפס הודעת שגיאה וביצוע התוכנית יופסק.

פונקצית השלם INT(X)
מקבלת את ערך המספר השלם הקרוב ביותר לערך משתנה הפונקציה X והקטן או שווה לערך X.

דוגמאות:

INT (5) יהיה 5
INT (5.8) יהיה 5
INT (—5.2) יהיה —6

דוגמאות לשימוש:

```
85 LET M2 = INT (M1 + 0.5)
```

המשתנה המספרי M2 יקבל את הערך המעוגל למספר שלם של המשתנה המספרי M1.

```
290 IF X/Y = INT(X/Y) THEN 100
```

אם ערך המשתנה המספרי Y הוא מחלק של ערך המשתנה המספרי X (כלומר X/Y מתחלק ללא שארית) אז התוכנית תסתעף להוראה 100. אחרת, תמשיך התוכנית להוראה הבאה.

פונקצית הסימן SGN(X)

מקבלת ערך 1 אם ערך משתנה הפונקציה X חיובי,
מקבלת ערך 0 אם ערך משתנה הפונקציה X אפס,
מקבלת ערך 1— אם ערך משתנה הפונקציה X שלילי.

דוגמא לשימוש:

```
120 ON SGN(A) + 2 GO TO 80, 190, 40
```

אם ערך המשתנה המספרי A שלילי, התוכנית תסתעף להוראה 80, אם ערכו אפס התוכנית תסתעף להוראה 190, ואם ערכו חיובי התוכנית תסתעף להוראה 40.

פונקצית הערך המוחלט ABS (X)

מקבלת את ערך המשתנה X אם ערכו חיובי או אפס,
מקבלת את הערך הנגדי (כלומר X—) אם ערכו שלילי.
לכן ABS(X) תקבל תמיד ערך חיובי או אפס.

דוגמאות לשימוש:

```
90 LET Y5 = ABS(Y5)
```

אם ערך המשתנה Y5 חיובי או אפס — ערכו לא ישתנה.
אם ערך המשתנה Y5 שלילי — ערכו ישאר אותו מספר אך בשינוי סימן (כלומר, הערך החדש של Y5 יהיה חיובי).

```
400 LET D7 = ABS(D1 — D2)
```

אם ערך המשתנים D1 ו־D2 מקיים D1>D2 אז המשתנה D7 יקבל את הערך D2 —D1. אם ערכם מקיים D1<D2 אז המשתנה D7 יקבל את הערך D1 — D2. כלומר, ערכו של D7 יהיה המרחק בין הערכים של D1 ו־D2.

פונקצית הערך האקראי RND(0)

מקבלת ערך מקרי בתחום הפתוח 1 —0. משתנה הפונקציה הנו תמיד אפס.

דוגמאות לשימוש:

```
110 LET  X = RND(0)
```

X יהיה מספר אקראי בין 0 ל־1.

```
350 LET Y = N*RND(0)
```

Y יהיה מספר אקראי בין 0 ל־N.

```
190 LET P2 = A + (B—A)*RND(0)
```

P2 יהיה מספר אקראי בין A ל־B ‏(A<B).

```
230 LET Q = INT(N * RND(0))
```

Q יהיה מספר אקראי שלם בין 0 ל־1— N ‏(N מספר שלם).

```
170 Z = M + INT ((N—M) * RND(0)
```

Z יהיה מספר אקראי שלם בין M ל־1— N ‏(M ; N מספרים שלמים, N>M).

פונקצית הקוסינוס COS(X)

מקבלת את הערך של סינוס X.

הערך של X חייב להיות ברדיאנים (כלומר $0 \leqslant X \leqslant 2\pi$).

פונקצית הקוסינוס COS(X)

מקבלת את הערך של קוסינוס X.

הערך של X חייב להיות ברדיאנים.

פונקצית הטנגנס TAN(X)

מקבלת את הערך של טנגנס X.

הערך של X חייב להיות ברדיאנים.

פונקצית הטנגנס ההפוכה ATN(X) (arctangent)

מקבלת את הערך ברדיאנים של הטנגנס ההפוך של X.

עבור X כלשהו מחזירה הפונקציה ערך בין $\pi/2$ — לבין $\pi/2$

פונקצית הלוגריתמוס הטבעי LOG(X) (Ln x)

מקבלת את הערך של הלוגריתמוס הטבעי של x (על בסיס e).

הערך של X חייב להיות חיובי.

פונקצית המעריך הטבעי EXP(X) (e)

מקבלת את הערך של e^x ‏(e = 2.71828).

מחרוזת (String)

בנוסף לטיפול במספרים, בקבועים מספריים, במשתנים מספריים ובביטויים אריתמטיים מאפשרת שפת ה־BASIC לטפל גם במחרוזות.

מחרוזת היא סידרת תווים כלשהי הסגורה בין סימני גרשיים (").

השימוש במחרוזות בתוכנית הנו לצורך הדפסת כותרות, שמות, הסברים, והודעות קבועות כלשהן. לא ניתן לבצע פעולות אריתמטיות על מחרוזות, מאחר והן חסרות כל משמעות מספרית או כמותית.

דוגמאות של מחרוזות:

"123"
"THIS IS A STRING CONSTANT"
"ABC123 DEF456"

משתנה מחרוזתי (String Variable)

זהו המשתנה המקבל "תוכן" של מחרוזת ויכול לשנות את "תוכנו" בזמן ביצוע התוכנית.

סימון משתנה מחרוזתי ב־BASIC נעשה על־ידי תו אלפביתי ואחריו (מימינו) תו הדולר ($).

לדוגמא:

B$
X$
A$

הערה: התוכנית מבחינה בין המשתנה A שהוא משתנה מספרי לבין המשתנה A$ שהוא משתנה מחרוזתי.

אין להשתמש במשתנים מחרוזתיים בביצוע פעולות אריתמטיות.

היחסים בין מחרוזות

אופרטורי־היחס המשמשים לקביעת יחסים בין משתנים מספריים, משמשים גם למחרוזות.

A$ = B$	פירושו שהמחרוזת A$ זהה למחרוזת B$.
A$<>B$	פירושו שהמחרוזות אינן זהות.
A$ < B$	פירושו שהמחרוזת A$ קודמת בסדר הלקסיקוגרפי (אלפביתי ו/או מספרי) למחרוזת B$.
A$ > B$	פירושו שהמחרוזת B$ קודמת בסדר הלקסיקוגרפי למחרוזת A$.

דוגמאות:

"AB" < "ABC"
"A1B3" < "AZB3" (מכיוון שהאותיות קודמות למספרים)
"ST" < "ST " (מכיוון שהמחרוזת הימנית ארוכה יותר — מכילה תווי רווח).

היחסים בין מחרוזות, ככל היחסים האחרים, הם ביטויים לוגיים בעלי ערך של אמת או שקר ויכולים לשמש בכל מקום בו מותר להשתמש בכאלה, למשל בהוראת התנאי וכד׳.

אין להשוות משתנים מספריים עם משתנים מחרוזתיים.

הוראות הקשורות לטיפול במחרוזות

השימוש במחרוזות מותר רק בחלק מהוראות ה־BASIC. הוראות אלה הן: READ, DATA, PRINT, LET, INPUT, ו־IF-THEN. להלן הכללים לשימוש בהוראות לצורך טיפול במחרוזות:

.READ ,DATA

הוראות אלה זהות להוראות READ ו־DATA שתוארו לעיל. בין המשתנים המופעים ברשימה שאחרי מלת המפתח READ יכולים להופיע גם משתני מחרוזות. בהכרזת ה־DATA שממנה יקראו ערכי המשתנים יופיעו מחרוזות במקומות המתאימים למשתנים המחרוזתיים.

דוגמא:

הוראת READ וההכרזה DATA ממנה יקראו הנתונים:

```
100 READ A,B,A$,X,R$
190 DATA 5,22,"EXAMPLE1", 3, "NUMBER"
```

בזמן ביצוע הוראה 100 המשתנים המספריים A, B ו־X יקבלו את הערכים 22,5 ו־3 בהתאמה, ולמשתנים המחרוזתיים A$ ו־B$ יוצב התוכן EXAMPLE1 ו־NUMBER, בהתאמה.

אם במקום מחרוזת (סידרת תווים בגרשיים) יופיע נתון כלשהו שאיננו סגור בגרשיים, המשתנה המחרוזתי יקבל את "הערך" הרשום, אך יתכן ויתווספו מספר תווי רווח לפני או אחרי התווים הרשומים. למשל, אם במקום ההכרזה 190 בדוגמא לעיל תהיה ההכרזה הבאה

```
190 DATA 5,22,1,3, "NUMBER"
```

אז המשתנה המחרוזתי A$ יקבל את "הערך" 1, ויתכן ויהיו בערך שיוצב ב־A$ מספר תווי רווח לפני או אחרי התו 1. במקרה זה התו 1 יוגדר על־ידי התוכנית כמחרוזת השייכת למשתנה המחרוזתי A$. התוכנית מבחינה בין התו 1 המוגדר כמחרוזת לבין התו 1 המוגדר כמספר, ושני התווים אינם זהים.

אם בהכרזת ה־DATA תופיע מחרוזת במקום בו צריך להיות קבוע מספרי, תודיע התוכנית על שגיאה וביצוע התוכנית יופסק. למשל אם במקום ההכרזה 190 בדוגמא לעיל תופיע ההכרזה

```
100 DATA 5, "EXAMPLE2",
```

אזי בזמן ביצוע ההוראה 100 תנסה התוכנית להציב את המחרוזת EXAMPLE2 למשתנה המספרי B, וכתוצאה מכן תווצר שגיאה והתוכנית תדפיס את הודעת השגיאה ILLEGAL DATA.

PRINT

הטיפול במחרוזות בהוראת ה־PRINT הוסבר קודם. הטיפול במשתנים מחרוזתיים יוסבר להלן.

אפשר לרשום ברשימת הקבועים, המשתנים המספריים והמחרוזות המופיעים אחרי הוראת ה־PRINT גם משתנים מחרוזתיים, כדי שתוכנם יודפס בזמן ביצוע ההוראה.

דוגמא:

```
250 PRINT A,B, "THIS IS" ; A$
```

בעת ביצוע הוראה זו תודפס שורה על גבי המסוף, אשר בה יופיעו, לפי הסדר, הנתונים הבאים: ערכו של המשתנה המספרי A, ערכו של המשתנה המספרי B, המילים THIS IS, ותוכנו של המשתנה המחרוזתי A$. למשל, אם ערכו של A הוא 5, ערכו של B 22, ותוכנו של A$ הוא EXAMPLE1, תודפס השורה הבאה על גבי המסוף:

```
5   22   THIS IS EXAMPLE1
```

LET

הוראת ההצבה LET מאפשרת, בנוסף לטיפול בקבועים מספריים, משתנים מספריים וביטויים אריתמטיים כפי שתואר לעיל, גם להציב תוכן למשתנה מחרוזתי.

דוגמאות:

```
120 LET P$ = "MONTH"
310 LET S$ = H$
```

בשורה 120 תוצב המחרוזת MONTH למשתנה המחרוזתי P$. בשורה 310 תוצב המחרוזת המיוצגת על־ידי H$ אל המשתנה המחרוזתי S$.

INPUT

הוראה זו מאפשרת לתוכנית לקבל נתונים מהמסוף כפי שהוסבר לעיל. המשתנים הרשומים אחרי הוראת ה־INPUT יכולים להיות משתנים מספריים או משתנים מחרוזתיים, ועל המשתמש בתוכנית להקיש על גבי המסוף את ערכם של המשתנים בהתאם לדרישת התוכנית.

דוגמא:

```
170 INPUT A,B$,A$,T
```

בהגיע התוכנית להוראה 170 יופיע על המסוף סימן שאלה (?) המודיע למשתמש שהתוכנית מצפה לנתונים. בהתאם להוראה זו, על המשתמש להקיש ערך מספרי עבור המשתנה A, מחרוזת עבור המשתנה המחרוזתי B$, מחרוזת נוספת עבור

המשתנה המחרוזתי A$, וערך מספרי עבור המשתנה המספרי T. הקשת הנתונים במסוף חייבת להיעשות בסדר המתאים לסדר הופעתם בהוראת ה־INPUT. בין כל שני נתונים חייב המשתמש להקיש פסיק (,). בזמן הקשת מחרוזת מספיק להפרידה מהנתונים האחרים על־ידי הקשת פסיקים לפניה ואחריה, ואין הכרח להקיש גרשיים.

דוגמאות:

```
50 INPUT A$
```

בהגיע המחשב להוראה זו יופיע התו ? על המסוף, המודיע למשתמש שהתוכנית מצפה לנתון. תשובת המשתמש תהיה, למשל:

```
? YES,
```

כלומר המשתמש יקיש את המחרוזת ואחריה התו פסיק, ולאחר מכן ילחץ על מקש המשלוח (carriage return).

```
60 INPUT A$,H,H$,R$
```

כאשר תגיע התוכנית לביצוע הוראה זו, יופיע התו ? על המסוף, ואז על המשתמש להקיש ארבעה נתונים: הראשון, השלישי והרביעי יהיו מחרוזות, והשני יהיה מספר. למשל —

```
? ONE, 5, FIVE, EIGHT,
```

שים לב שאחרי המחרוזת האחרונה הוקש פסיק.
אם התוכנית מצפה לקבל מספר והמשתמש מקיש מחרוזת, תופיע הודעת השגיאה ILLEGAL DATA.

IF-THEN

בהוראה זו ניתן להשתמש במחרוזות ובמשתנים מחרוזתיים בדומה לשימוש שתואר עבור קבועים מספריים, משתנים מספריים או ביטויים אריתמטיים, אך יש לזכור שהתנאי חייב להיות לגבי ערכים מספריים בלבד או לגבי מחרוזות בלבד, ולא ניתן לערב ביניהם.

דוגמאות:

```
80 IF A$ = "YES" THEN 300
110 IF P$ = Q$ THEN 40
190 IF H$ < "POR" THEN 420
230 IF R$ >= S$ THEN 200
```

בכל אחד מהמקרים לעיל תתבצע ההסתעפות רק אם התנאי המופיע בהוראה מתקיים, אחרת — ימשיך ביצוע התוכנית להוראה הבאה.

מערכים (Arrays)

מערך קווי הנקרא וקטור (Vector)

מערך קווי הוא למעשה סידרה בעלת אינדקס (index או subscript) אחד. לוקטור ניתן שם על־ידי שימוש בשם של משתנה מספרי, או משתנה מחרוזת (לפי העניין) ואחריו (בסוגריים) האינדקס.

הצורה היא, לדוגמא:

```
A( )
A$( )
```

בתוך הסוגריים יכול לבוא ביטוי אריתמטי כלשהו. ערך ביטוי זה יקוטע למספר שלם שהוא יהיה האינדקס של הוקטור.

איברי הוקטור הם, לפי הסדר:

A(1),A(2),A(3),........,A(N),....

האיבר A(N) משמש כמשתנה מספרי לכל דבר ומותר להציב בו ערכים מספריים על־ידי הוראות הצבה וחישוב. כל הכללים החלים על משתנים מספריים נשמרים גם עבור איברי מערך מספרי. האיבר A$(N) משמש כמשתנה מחרוזתי לכל דבר, ומותר להציב בו מחרוזות ולהשתמש בו בכל ההוראות המתאימות למשתנים מחרוזתיים.

דוגמאות:

```
110 LET A(3) = 2*P + 1
240 LET A$(6) = "XY"
```

במקרה הראשון, האיבר השלישי של הוקטור המספרי A יקבל את הערך שיחושב לביטוי 2P + 1.
במקרה השני, האיבר השישי בוקטור המחרוזתי A$ יקבל את התוכן XY.
האינדקס של וקטור משמש אפוא **כתובת** של התוכניתן, המאפשר לפנות לכל איבר בוקטור.

מערך מלבני הנקרא מטריצה (Matrix)

מערך מלבני הוא "מלבן" של מקומות, שכל מקום בו מאופיין על־ידי שני אינדקסים:

אינדקס השורה	(Row subscript)
אינדקס העמודה	(Column subscript)

גם למטריצה ניתן שם המסומל על־ידי סמל משתנה מספרי או משתנה מחרוזתי (לפי העניין) ואחריו **בסוגריים** יבואו שני האינדקסים כשביניהם פסיק: הראשון

הוא אינדקס השורה והשני אינדקס העמודה. כמו בוקטור, יכולים האינדקסים להיות ביטויים אריתמטיים כלשהם.

הצורה של איבר המטריצה תהיה:

```
A$( , )
 A( , )
```

מלבן איברי המטריצה בעלת M שורות ו־N עמודות נראה כך:

```
A(1,1) A(1,2) A(1,3) ...... A(1,N)
  .
  .
  .
A(2,1) A(2,2) A(2,3) ...... A(2,N)
  .
  .
  .
A(M,1) A(M,2) A(M,3) ...... A(M,N)
```

וקטור הנו מקרה פרטי של מטריצת בעלת עמודה אחת.

דוגמאות:

```
310 LET A(3,5) = 3*X+X**2
340 LET A(I+3,2*J) = 9
270 LET A(R,12) = A(R*3,10) * A(2,R+1)
200 IF A$(8,S+2) = B$(1,6) THEN 290
180 READ A$(1,2), A(1,2), A(1,3), B(2,1)
```

DIM

הכרזת המימד DIM מגדירה את מימדי המטריצות בתוכנית (מימד המטריצה פירושו מספר השורות ומספר העמודות במטריצה).

צורת ההוראה:

לוקטור: <LN> DIM <V> (<N>)

למטריצה: <LN> DIM <V> (<M>, <N>)

וכאן חייבים <N>, <M> להיות מספרים שלמים חיוביים.

ניתן באותה שורה להכריז על מימדי מספר כלשהו של וקטורים ומטריצות.

למשל:

```
10 DIM X(100), Y(20,50), A$(5), Z$(2,2)
```

כל וקטור או מטריצה שנעשה בהם שימוש בתוכנית חייבים להופיע בהוראת המימד, בה נקבע מספר השורות ו/או העמודות המקסימלי שלהם.

ההכרזה יכולה לבוא בכל מקום בתוכנית, אולם רצוי מאד להכריז על כל הוקטורים והמטריצות בראש התוכנית (ולפחות לפני השימוש בהם בתוכנית).

אם במהלך ביצוע התוכנית יחרוג אחד האינדקסים של וקטור או מטריצה מהחסמים שלו: התחתון — 1 והעליון כפי שנקבע בהכרזת ה־DIM, תופיע הודעת שגיאה בנוסח:

SUBSCRIPT OUT OF RANGE

וביצוע התוכנית ייפסק.

במשך התכנית נתן לשנות את מימדי המטריצה, בתנאי שהמימדים החדשים אינם גדולים מאלו המצוינים בהוראות ה־DIM.

MAT READ ; MAT PRINT

באמצעות ההוראות MAT PRINT, MAT READ ניתן לקרוא ולהדפיס מטריצות ללא פירוט כל איבר ואיבר של המטריצה.

לדוגמא:

```
100 MAT READ A,F,H,G
150 MAT PRINT C
175 MAT READ Z
190 MAT PRINT A,L
```

שורה 100 מורה לתוכנית לקרוא את המטריצות H,F,A, ו־G בהתאם למימדים שצוינו קודם לכן בהכרזת ה־DIM. הנתונים נקראים לפי השורות, היינו:

A(1,1), A(1,2), A(1,3) ...... A(2,1), A(2,2) ......

ובסדר זה צריכים הנתונים להיכתב בכרטיסי ה־DATA. התוכנית תקרא נתונים עד למילוי המטריצה. אין אפשרות לקרוא או להדפיס רק חלקים של מטריצה.

דוגמא: קטע תוכנית המטפל בשתי מטריצות.

```
110 DIM L(2,3), M(2,2)
150 MAT READ L,M
160 LET L(2,2) = —2*L(2,2)
200 MAT PRINT L,M
500 DATA 1,2,3,4,5,6,3,—12,0,7
```

שורה 110 מגדירה את L כמטריצה 3×2 ואת M כמטריצה 2×2. שורה 150 קוראת את נתוני ה־DATA בשורה 500 ומציבה אותם במטריצות; קודם לכן נקראים איברי המטריצה L (ששה איברים), ולאחר מכן נקראים איברי המטריצה M

(ארבעה איברים). האיבר L(2,2) מחושב מחדש בשורה 160. שתי המטריצות מודפסות וערכן:

$$L = \begin{pmatrix} 1 & 2 & 3 \\ 4 & -10 & 6 \end{pmatrix} \qquad M = \begin{pmatrix} 3 & -12 \\ 0 & 7 \end{pmatrix}$$

חיבור, חיסור והכפלת מטריצות

בנוסף לפעולות אריתמטיות על איברי המטריצות (אשר, כאמור, ניתן לבצע עליהן כל פעולה אריתמטית המותרת לביצוע במשתנים מספריים ובאותם כללים, ניתן לבצע גם פעולות על מטריצות שלמות. ההוראות המותרות לביצוע על מטריצות שלמות הן:

חיבור	MAT C = A + B
חיסור	MAT C = A — B
הכפלה	MAT P = Q * R

בהוראות אלה המשתנים המספריים R,Q,P,C,B,A מייצגים מטריצות אשר הוגדרו לפני ביצוע הפעולות, ומימדיהן ידועים לתוכנית.

ניתן לחבר, לחסר ולהכפיל מטריצות באמצעות ההוראות הללו בתנאי שמימדי המטריצות מתאימים. אי התאמה גורמת להפסקת ביצוע התוכנית והדפסת הודעת שגיאה מסוג:

DIMENSION ERROR

בכל הוראה ניתן לבצע רק פעולה אחת. פעולות על מטריצות אינן יכולות להופיע בתוך ביטויים אריתמטיים פרט לביטויים המוגדרים כאן.

הכפלה סקלרית

פעולה נוספת אשר מותרת לביצוע על מטריצה שלמה.

ההוראה:

```
MAT X = (<exp>) * D
```

כאשר <exp> הוא ביטוי אריתמטי (שתוצאתו מספר) גורמת להכפלת המטריצה D במספר והכנסת התוצאה למטריצה X.

גם בהוראה זו רק פעולה אחת מותרת לכל שורה.

לדוגמא:

```
100 MAT F = (2) * G
150 MAT Q = (2.33 + M) * Q
750 MAT B = (N) * A
```

כאן המשתנים המספריים G, Q ו־A מייצגים מטריצות שמימדיהן ידועים לתוכנית, M ו־N הנם משתנים מספריים (שאינם מטריצות).

מטריצת היחידה

מוגדרת על־ידי אחד מהביטויים הבאים:

```
MAT B = IDN
MAT R = IDN(<exp1>, <exp2>)
```

הוראה זו מציבה אפסים לכל איברי המטריצה פרט לאיברי האלכסון הראשי, שאליהם מציבה התוכנית את המספר 1. מותר להשתמש בהוראה זו רק למטריצות ריבועיות (כלומר מספר השורות שווה למספר העמודות).
בשורה הראשונה מטריצה B מוגדרת כמטריצה יחידה, בהתאם למימדיה שהוגדרו קודם לכן. אם לא הוגדרה כריבועית, תתקבל שגיאת מימד
(MATRIX DIMENSION ERROR)

בשורה השנייה, מימדי המטריצה B נקבעים מחדש תוך כדי הגדרתה כמטריצת יחידה; המימדים החדשים חייבים להיות קטנים או שווים מהמימדים שנקבעו למטריצה בהכרזת ה־DIM.
ברור שערכי <exp1> ו־<exp2> חייבים להיות שווים, ואם אינם שלמים תקטע התוכנית את החלק העשרוני של התוצאה ותשמש לצורך קביעת המימדים רק בחלק השלם.

דוגמאות:

```
190 MAT A = IDN
100 MAT V = IDN(2*N+1, 2*N+1)
120 MAT B = IDN(Q,Q)
```

בדוגמאות אלה, המשתנים המספריים B,V,A הוגדרו כמטריצות בהכרזת DIM, N ו־Q הנם משתנים מספריים (שאינם מטריצות).

מטריצה מוחלפת

ניתן לקבל את A^t (transpose A) באמצעות ההוראה:

```
MAT Y = TRN(A)
```

ההוראה MAT A = TRN(A) אסורה! אסור להפוך את המטריצה ועדיין לשמור על אותו סימון.
מימדי המטריצות בהוראה חייבים להיות זהים (כלומר המימדים של Y זהים למימדים של A בהתאם להגדרתם האחרונה).

לדוגמא:

```
400 MAT U = TRN(V)
300 MAT G = TRN(H)
```

בדוגמאות אלה, G ו־H הן מטריצות מאותו מימד, U ו־V אף הם מטריצות מאותו מימד.

מטריצה הופכית

הוראה:

MAT I = INV(J)

מגדירה את המטריצה I כמטריצה ההופכית של המטריצה J, J^{-1}.

הוראה: MAT I= INV(I) אסורה! (כלומר מטריצה איננה יכולה להיות ההופכית של עצמה).

מתקבלת הודעת שגיאה וביצוע התוכנית מופסק במידה והמטריצה סינגולרית (כלומר לא קיימת לה הופכית).

גם בהוראה זו מימדי המטריצות בהוראה חייבים להיות זהים בהתאם להגדרתם האחרונה.

מטריצת אפסים

הוראה:

MAT C = ZER

גורמת לאיפוס כל איברי המטריצה C.

הוראה:

MAT C = ZER(<EXP1>, <EXP2>)

משנה את מימדי המטריצה C למימדים חדשים — הערך של הביטוי EXP1 יהיה מספר השורות החדש, הערך של EXP2 יהיה מספר העמודות החדש, בתנאי שמימדים אלה קטנים או שווים למימדים שנקבעו למטריצה C בהכרזת המימד DIM. אם EXP1 או EXP2 אינם מספרים שלמים, קוטעת התוכנית את החלק העשרוני של המספר ומשתמשת רק בחלק השלם לקביעת המימד.

מטריצת האפסים איננה חייבת להיות מטריצה ריבועית.

מטריצת אחדים

הוראה:

MAT C = CON

קובעת את כל איברי המטריצה C לערך 1.

ניתן להגדיר מימד באמצעות ההוראה —

MAT C = CON(M,N)

כל האמור לעיל לגבי מטריצת האפסים נכון למטריצת האחדים.

תוכנית לדוגמא

תוכנית זו מבצעת חישובי ערך עתידי עבור סכום C, בשערי ריבית R שונים, ועבור עשר תקופות.

בתוכנית נעשה השימוש בהוראות MAT ו־DEF FN.

הוראה 40 מגדירה פונקציה המעגלת את המשתנה Y (משתנה הפונקציה) לשלוש ספרות עשרוניות.

ההוראות 70 עד 110 בונות מטריצה המכילה את הערכים העתידיים עבור C = 1.

ההוראה 130 מקבלת את הערך הרצוי של C באמצעות המסוף.

ההוראה 220 יוצרת את המטריצה הסופית, המכילה את הערכים העתידיים של סכום C.

```
10 REM PROGRAM TO COMPUTE THE
20 REM TABLE   OF FUTURE VALUE OF C I.L.
30 DATA 0.06,0.12,0.18,0.24
40 DEF FNX(Y) = INT(0.5+1000*Y)/1000
50 DIM R(4),U(10,4),W(10,4)
60 MAT READ R
70 FOR N = 1 TO 10
80 FOR J = 1 TO 4
90 LET U(N,J) = FNX((1+R(J))**N)
100 NEXT J
110 NEXT N
120 PRINT "TYPE VALUE OF C"
130 INPUT C
140 PRINT
150 PRINT "FUTURE VALUE OF "; C;"I.L."
160 PRINT
170 FOR J = 1 TO 4
180 PRINT "R="; R(J),
190 NEXT J
200 PRINT
210 PRINT
220 MAT W = (C)*U
230 MAT PRINT W
240 END
```

על המסוף יתקבל:

```
TYPE VALUE OF C
?10
FUTURE VALUE OF 10 I.L.
```

R= .06	R= .12	R= .18	R= .24
10.6	11.2	11.8	12.4
11.24	12.54	13.92	15.38
11.91	14.05	16.43	19.07
12.62	15.74	19.39	23.64
13.38	17.62	22.88	29.32
14.19	19.74	27.	36.35
15.04	22.11	31.85	45.08
15.94	24.76	37.59	55.9
16.89	27.73	44.35	69.31
17.91	31.06	52.34	85.94

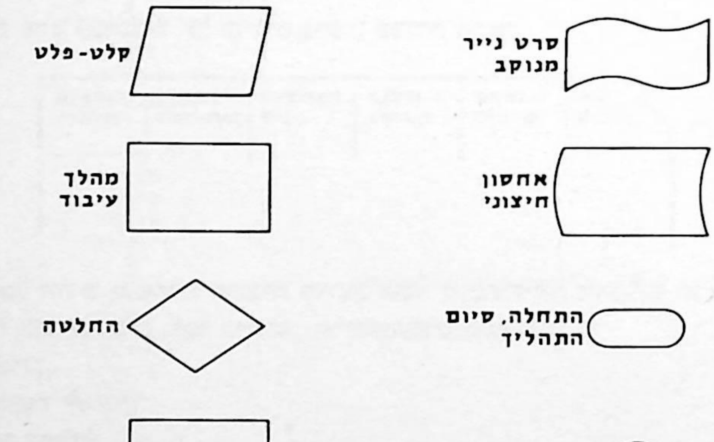
3.

תרשימי זרימה, תרשימי תהליך

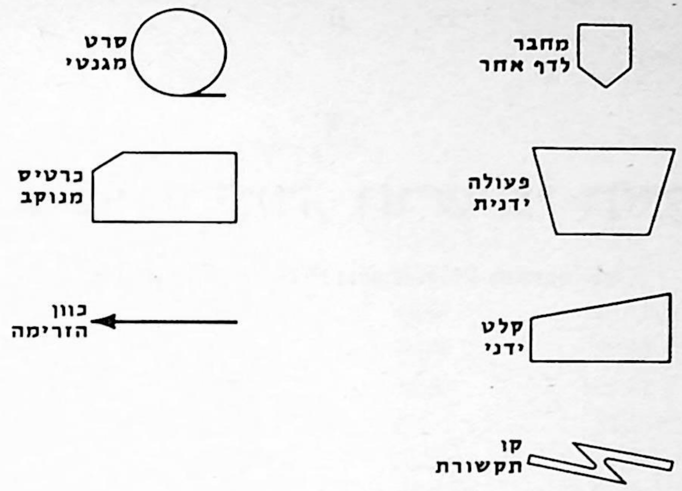
(Program Flowchart, Process Flowchart)

תרשים זרימה הוא תרשים סכמטי של מהלך ביצוע ההוראות בתוכנית, בו מיוצגת כל הוראה על־ידי סימן מוסכם. תרשים תהליך הוא תרשים סכמטי של מהלך ביצוע שלבי העיבוד בתהליך עיבוד הנתונים, כאשר כל סימן מייצג מהלך עיבוד (תוכנית שלמה), קובץ, או פעולת קלט/פלט. (דוגמא לתרשים תהליך ראה, למשל, תרשים 1.44 בעמוד 71, ותרשים 1.45 בעמוד 72).
קיימים סימנים מקובלים המשמשים להכנת תרשימי זרימה ותרשימי תהליך.

סימנים אלה הם:



תרשים 2.4:



דוגמאות לשימוש בתרשימי זרימה:

דוגמא 1 — הוצאות סוכן הנסיעות (תרשים 2.5)

חברת אבג דורשת מ־55 סוכני המכירות שהיא מעסיקה למסור את הנתונים הבאים:

מספר הימים ששהו מחוץ למשרד

הוצאות נסיעה

הוצאות לינה

הוצאות לארוחות

הוצאות אחרות:

נתונים אלה מנוקבים על כרטיס ניקוב בצורה הבאה:



בסוף כל חודש מופעלת תוכנית מחשב אשר מבצעת את הדברים הבאים:

1. קוראת את הנתונים מכל כרטיס; פריטי הקלט כוללים:

   שם הסוכן

   ימים מחוץ למשרד

   הוצאות נסיעה

   הוצאות לינה

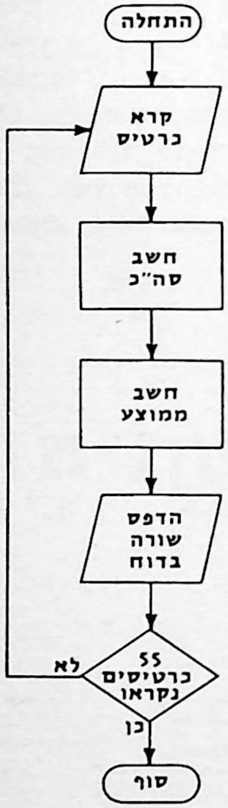
   הוצאות לארוחות

   הוצאות אחרות

2. מעבדת את הנתונים.
העיבוד כולל:
חישוב סה"כ = נסיעה + לינה + ארוחות + אחרות

$$\text{חישוב ממוצע הוצאות יומי} = \frac{\text{סה"כ}}{\text{ימים}}\ .$$

3. מדפיסה דוח, הכולל שורה לכל סוכן, ובו יופיע:
שם הסוכן
סה"כ ההוצאות לחודש
ממוצע הוצאות יומי
ימים מחוץ למשרד
4. לאחר שנקראו כל 55 הכרטיסים, מסתיים ביצוע התוכנית.



תרשים 2.5: תרשים זרימה לחישוב הוצאות סוכן הנסיעות

דוגמא 2 — עמלת מכירות (תרשים 2.6)

חברת "דהו" מבצעת את חישובי העמלה לסוכניה לפי התוכנית המתוארת להלן: הסוכנים מקבלים עמלה (COMM) לפי אחוז מהמכירות. אחוז העמלה (RATE)

תלוי בסוג הפריט שנמכר. קיימים שלושה סוגי פריטים אשר נמכרים על־ידי חברת "דהו":

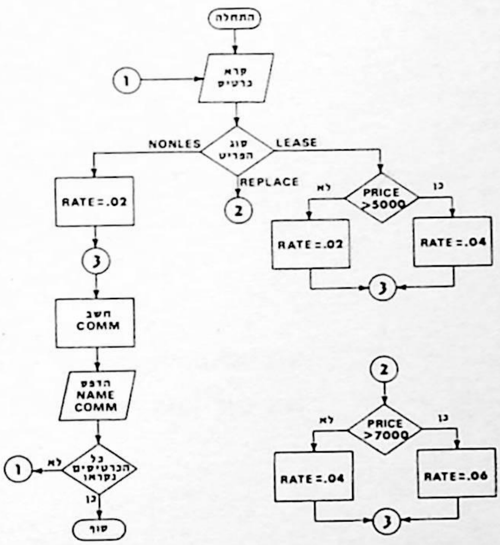
1. פריטים אשר ניתנים להשכרה (LEASE).
2. פריטים אשר לא ניתנים להשכרה (NONLES).
3. פריטים להחלפה (REPLACE).

אם הפריט הוא מסוג LEASE ומחירו מעל ל־5000 ל"י העמלה (RATE) היא 4% ממחירו. אם מחירו נמוך מ־5000 ל"י העמלה היא 2%.
אם הפריט הוא מסוג NONLES, העמלה היא 2%.
אם הפריט הוא מסוג REPLACE, העמלה היא 6% אם המחיר הוא מעל ל־7000 ל"י, ו־4% אם מחירו נמוך מ־7000 ל"י.

כרטיסי הקלט מכילים את הנתונים הבאים:

שם הסוכן	—	NAME
סוג הפריט	—	LEASE, NONLES, REPLACE
מחיר המכירה	—	PRICE (בלירות).

התוכנית מחשבת את העמלה (COMM) לזכות כל אחד מהסוכנים, ומדפיסה דוח הכולל שורה לכל מכירה שבוצעה, ובכל שורה שם הסוכן (NAME) והעמלה (COMM) העומדת לזכותו הנובעת מהמכירה. העמלה מתקבלת על־ידי הכפלת אחוז העמלה במחיר המכירה (COMM = RATE • PRICE).



תרשים 2.6: תרשים זרימה לחישוב עמלת המכירות.

בדוגמא זאת נעשה שימוש במחברים כדי לקשור בין צעדים שונים בתרשים הזרימה.

III. חלק שלישי

מערכות מידע

1.
הגדרה, תכונות ומבנה של מערכת מידע

הגדרתה של מערכת מידע

למערכת מידע מספר רב של הגדרות, זאת משום שהנושא מורכב וההסתכלות בו יכולה להיעשות מנקודות מבט שונות.

אנו נבחר בהגדרה הבאה:
מערכת המפיקה מידע, שמטרתו תמיכה בתפעולו ובניהולו של הארגון. ההגדרה לא תהיה בהירה וברורה די הצורך אם לא נפרט את המשאבים שבהם היא עושה שימוש. המערכת עושה שימוש בחומרה ובתוכנה של המחשב, בנהלים ידניים, במודלים של החלטה וניהול ובמאגרי נתונים.
הגדרה זאת אינה כללית, משום שהיא מגדירה **מערכת מידע ממוחשבת**, שאינה יכולה להיות מופעלת ללא מחשב. ואמנם אלו הן המערכות שאנו עוסקים בהן. אולם המחשב, עם היותו אמצעי עיקרי, אינו האמצעי הבלעדי, לאמור: עוסקים אנו במערכת אדם־מחשב.
גורם עיקרי נוסף, המאפיין את מערכות המידע שבהן אנו עוסקים הוא, שהמידע שהן מפיקות תומך הן בתפעולו והן בניהולו של הארגון. מושגים אלה יובהרו להלן.
הנושאים שיש לדון בהם כדי להבין את מהותה, תפקידה ומבניה של מערכת מידע ואלה הם:

— גישת המערכות
— מערכת הארגון
— מערכת הניהול
— מבנה של מערכת מידע
— גישות לבניית מערכות מידע בארגון.

גישת המערכות

גישת המערכות עיקרה ראיית השלם המורכב בסביבתו מבלי לאבד בו־זמנית את ראיית מרכיביו של אותו שלם, או לשון אחר, היכולת לטפל במרכיב (תת־מערכת) תוך ראיית אותו מרכיב כחלק ממכלול שהוא משתייך אליו.

גישת המערכות היא אוניברסלית מחד, היינו דרך מחשבה ופילוסופיה של ראיית דברים, ומאידך משמשת כלי לניתוח בעיות, מצבים וגופים מורכבים.

מהי מערכת ?

מערכת היא קבוצת גורמים היוצרים שלמות מורכבת אחת.

קבוצת הגורמים יוצרת מערכת רק כאשר הגורמים הבונים אותה קשורים זה לזה וקיימת ביניהם תלות ופעילות הדדית. מערכת תהיה אפוא מורכבת מלפחות שני מרכיבים עם יחס גומלין ביניהם. אוסף מרכיבים ללא פעילות גומלין ביניהם אינו מערכת. משום היות המערכות שבהן אנו עוסקים מערכות העשויות בידי אדם (בניגוד למערכות טבעיות), יש למערכות הללו מטרה (או מטרות). מרכיבי המערכת יכולים להיות בעצמם מערכות ואז הם ייקראו תת־מערכות ואילו המערכת שבה עוסקים יכולה להיות מרכיב במערכת־על (suprasystem or metasystem). אותה דרך ניתוח טובה לתת־מערכת, מערכת ומערכת־על.

ניתוח מערכת (כל מערכת) מתבצע על־ידי איפיונה. פעולת האיפיון הנה קביעת ערכיהם של המשתנים העיקריים הבונים את המערכת.

להלן פירוט המשתנים העיקריים הללו :

א. גבולות המערכת

גבולות מערכת הנם אותו אזור המפריד את המערכת מסביבתה וממערכות אחרות.

לדוגמא : הגבול של מערכת הטלפון המקומית הוא כל הטלפונים המחוברים למרכזת מסוימת אחת ; גבול מערכת חברתית הוא קבוצת אילוצים המונעים חלק מהתיקשורת הפנימית שלה להיות זמינה לסביבתה, כולל מערכת העל שלה היא שייכת.

בדרך כלל יש פחות מעבר של חומר, אנרגיה ומידע מעבר לגבולות המערכת מאשר בתוך המערכת.

שתי הדוגמאות ממחישות את ההבדל בין גבולות פיסיים לבין גבולות קונספ־טואליים־לוגיים.

הגבולות תוחמים את המערכת. קביעת הגבולות אינה, בכל מקרה, פעולה פשוטה וישירה. פעמים, התהליך איטרטיבי (חוזר אחורה לשלבים מוקדמים) ויש בו מידה רבה של שרירותיות. התיחום מתבצע על סמך המידע הזמין בעת ביצוע הפעולה וכל הניסיון והידע שיש למבצע התיחום.

המושג גבול־מערכת חשוב לתיכון (design) של מערכות מידע. ההכרה בצורך

התיחום כופה על המתכננים לשקול בחירה בין היקף לבין מורכבות ובין משתנים כרי־שליטה לבין משתנים שאינם ברי־שליטה.

ב. המטרות, היעדים ומדדי הביצוע של המערכת

מערכות עשויות בידי אדם הנן בדרך כלל מכוונות מטרה (purposeful systems). הבנה והגדרה נכונה של מטרות המערכת שמנתחים אותה הנה תנאי ראשון להצלחת ניתוחה. אולם לא די בהבנה ובהגדרה של המטרות, שהן מטבען כלליות. יש להבין ולהגדיר גם את היעדים (שהם תרגום של המטרות הכלליות) וגם את מדדי הביצוע לפיהם שופטים את טיב תיפקודה של המערכת להשגת יעדיה ומטרותיה. מדדים כאלה יכולים להיות, לדוגמא, היחס בין ההכנסות להוצאות או היחס בין ההכנסות בפועל להכנסות הפוטנציאליות.

ג. קלטים ופלטים, אל המערכת וממנה

המערכות שבהן אנו עוסקים הנן מערכות פתוחות שקשרי גומלין להן עם סביבתן אשר מתבטאים בקלטים הנכנסים מהסביבה אל המערכת ובפלטים היוצאים מהמערכת אל סביבתה. קלטים ופלטים אלה, או במלים אחרות, תשומות ותפוקות, יכולים ללבוש כל צורה שהיא: חומר, אנרגיה או מידע. מערכות ארגוניות שהן מערכות חברתיות הנן תמיד מערכות פתוחות.

ד. גורמי המערכת

פירוקה של המערכת לגורמיה היא פעולה הדומה לפעולת תחימת המערכת בין גבולות. פעולה זו אינה תמיד חד־משמעית, ישירה ופשוטה. היא תלויה במטרת הניתוח, טיבו, ניסיונו ונקודת מבטו של המנתח ויש בה אפוא מידה מסוימת של שרירותיות.
פירוק מערכת לגורמיה יכול להעשות ברמות סיכום שונות. הרמה תיקבע על־ידי מטרת הניתוח וכן היקפה ומורכבותה של המערכת. ניתוח כל מרכיב שהוא תת־מערכת ייעשה בדרך שהותוותה למערכת עצמה.

לגבי כל מרכיב ומרכיב במערכת יש לקבוע את:

— היעדים והקשר שלהם למטרות המערכת שלה הם שייכים.
— התהליכים המתבצעת במרכיב (הטרנספורמציה).
— המשאבים שבהם הוא עושה שימוש.
— מקומו במידרג המרכיבים.

באותם מקרים שאין מנתחים בהם את המבנה הפנימי של המרכיבים, יש להתיחס אל המרכיב כאל "תיבה שחורה". התייחסות כזו פירושה ניתוח והבנת הקלטים והפלטים אל המרכיב וממנו ללא ידיעת המבנה והתהליכים המבצעים את הטרנספורמציה של הקלט לפלט בתוך המרכיב. שיטה זו יעילה כאשר עוסקים במערכות מורכבות בעלות מספר רב של מרכיבים וההתעניינות מתמקדת רק

בחלק מהמרכיבים, או כאשר נמצאים בשלב מוקדם של פעולת הניתוח בעוד מבנה המערכת הוא המטרה של הניתוח. רק בשלבים מתקדמים יותר של הניתוח, כאשר המבנה מוגדר ומנותח יורדים אל המבנה הפנימי של מרכיבי המערכת.

יש להבחין בין שני סוגי מרכיבים: אלה העוסקים בטרנספורמציה של חומר ואנרגיה ואלה העוסקים בטרנספורמציה של מידע. ראוי לציין שהמרכיבים העוסקים בטרנספורמציה של מידע עושים שימוש בחומר ואנרגיה כדי למלא את יעודם (למשל: ציוד מחשב).

ה. קשרי הגומלין בין גורמי המערכת

קשרי הגומלין בין גורמי המערכת הם ההופכים כאמור את קבוצת המרכיבים למערכת.

קשרי גומלין אלה הם אחד הגורמים העיקריים למורכבותו של הניתוח, משום שניתוח המערכת פירושו ניתוח מאפייני המערכת כיחידה שלמה, ניתוח מרכיביה וקשרי הגומלין בין המרכיבים.

הקשר בין המרכיבים מתבצע באמצעות מימשקים (interfaces) שהם קלט ופלט היכול ללבוש צורה של חומר, אנרגיה או מידע.

יש לזכור שקיימים גם קשרי־גומלין בין מרכיבי המערכת לבין סביבתה וגם את אלה יש להגדיר ולנתח.

רשת התקשורת בתוך המערכת מהווה חלק מקשרי הגומלין בין המרכיבים וזאת כשמדובר בקשרי גומלין של מידע. מרכיבי המערכת וקשרי הגומלין ביניהם נקראים **מבנה המערכת**. טיב הביצוע של המערכת הנו, בין השאר, פועל יוצא של טיב המבנה שלה. בונה מערכת המידע עושה שימוש בגישת המערכות כשהוא קובע את מבנה המערכת תוך צירוף פעילויות למרכיב אחד, קביעת המבנה המידרגי והמימשקים בין המרכיבים.

ו. סביבת המערכת

כללית ניתן להגדיר את סביבת המערכת כל שנמצא מחוץ לגבולותיה. אולם הסביבה, בהתאם להגדרה זאת, היא אין־סופית. לפיכך, סביבתה של אותה מערכת אשר מנתחים היא הסביבה הרלוואנטית לה, לאמור: הסביבה שעמה יש למערכת קשרי־גומלין. את הסביבה הרלוואנטית ניתן לחלק לסביבה קרובה ולסביבה רחוקה.

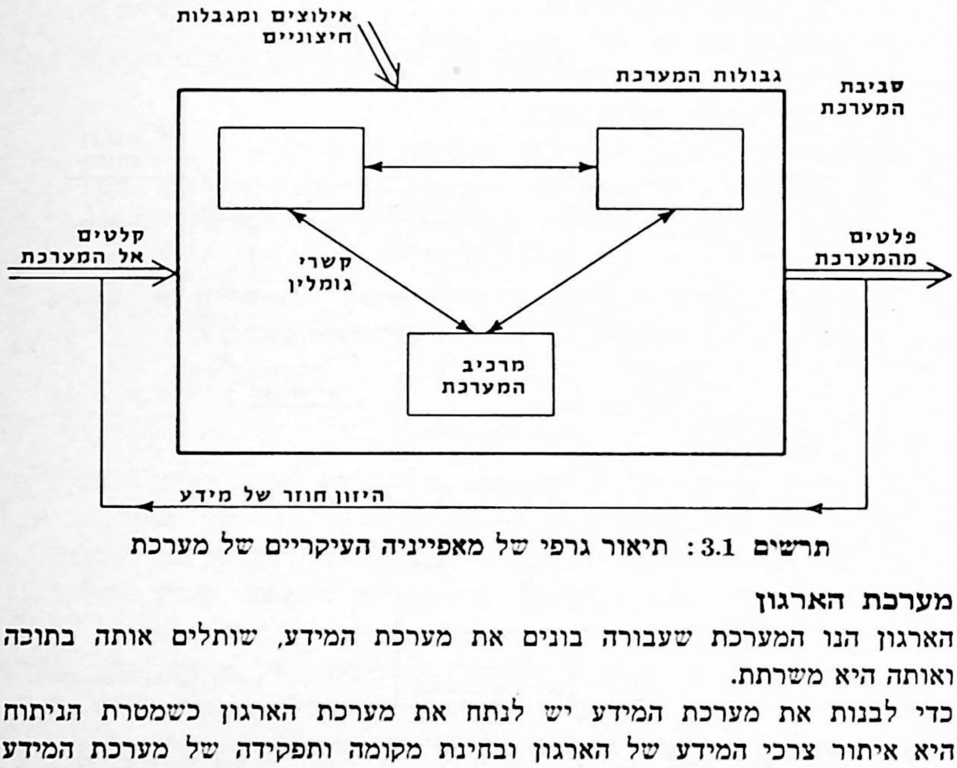
ז. מגבלות ואילוצים הפועלים על המערכת

על כל מערכת פועלים מגבלות ואילוצים אשר יש לאתרם, לזהותם ולנתחם. יש להבחין בין מגבלות ואילוצים הפועלים על המערכת על־ידי סביבתה הרלוואנטית לבין מגבלות ואילוצים הפועלים בתוך המערכת כפועל יוצא ממדיניותה, מבניה ותכונותיה של המערכת.

ח. הגורם המבקר או תת־מערכת הבקרה

תת־מערכת הדרושה לכל מערכת מכוונת־מטרה היא תת־מערכת הבקרה. זהו אותו מרכיב המקבל מידע מכל גורמי המערכת ומסביבתה הרלוואנטית, מקבל החלטות ומעביר הוראות שביצוען מאפשר תיפקוד יעיל, נכון ומכוון־מטרה של המערכת ושומר על היות פלט המערכת בין גבולות רצויים.

במערכת האדם — המוח הנו הגורם המבקר. בארגונים — זוהי מערכת הגיהול המתפקדת על בסיס המידע הבא אליה כהיזון־חוזר והמתאר את פעולת המערכת (היזון חוזר פנימי) ועל בסיס מידע הבא אליה כהיזון־חוזר מהסביבה הרלוואנטית (היזון חוזר חיצוני). יכולתה של מערכת להגיב להפרעות תלויה בטיבו של המידע הבא כהיזון חוזר, בעיכובים בתימסורת שלו ובתהליך הבקרה המתאים את משתני המערכת כתשובה לאותות ההיזון החוזר (המידע).



תרשים 3.1: תיאור גרפי של מאפייניה העיקריים של מערכת

מערכת הארגון

הארגון הנו המערכת שעבורה בונים את מערכת המידע, שותלים אותה בתוכה ואותה היא משרתת.

כדי לבנות את מערכת המידע יש לנתח את מערכת הארגון כשמטרת הניתוח היא איתור צרכי המידע של הארגון ובחינת מקומה ותפקידה של מערכת המידע שתספק צרכים אלה.

הארגון הנו מערכת ממשית, חברתית, והוא כולל גם מרכיבים פיסיים ומופשטים. תפקידו המוגדר של הארגון הוא הפיכת משאבים (על־ידי טרנספורמציה) למוצרים, שירותים או רעיונות, ושילובם.

המשאבים העוברים טרנספורמציה הם הקלט למערכת ואילו המוצרים, השירותים, או הרעיונות הם פלטי המערכת, שתפקידם השגת מטרות המערכת.

מרכיבי מערכת הארגון

כל ארגון, ברמת מורכבות והיקף כלשהם, כולל שלושה סוגי מרכיבים (מרכיב אחד מכל סוג לפחות):

— מרכיב התפעול

תפקידו ביצוע טרנספורמציה של משאבים למוצרים, שירותים או רעיונות. זהו המרכיב העיקרי של הארגון משום שהוא משיג את מטרותיו.

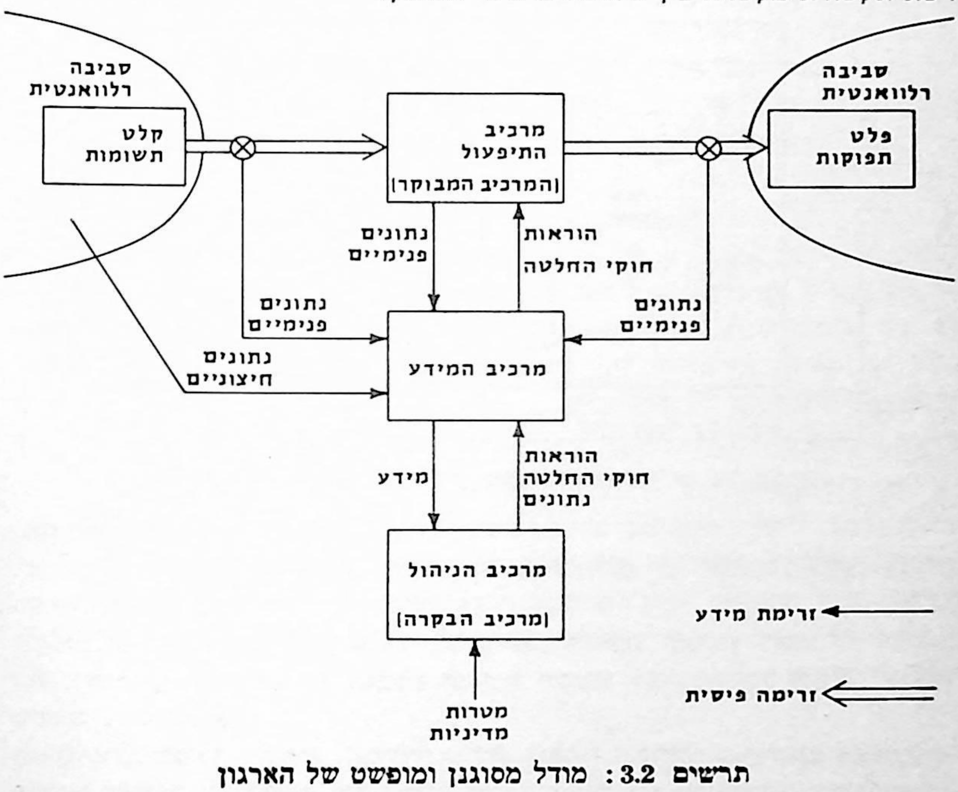
— מרכיב הניהול

תפקידו ביצוע פעולת הבקרה של מרכיבי התפעול (שהם המרכיב המבוקר).

— מרכיב המידע

תפקידו, מחד, להעביר למרכיב התפעול את ההוראות שמקורן במרכיב הניהול, ומאידך, למסור למרכיב הניהול מידע על המתרחש במרכיב התפעול (היזון חוזר של מידע).

רשת תקשורת מקשרת בין שלושת מרכיבי הארגון.



תרשים 3.2: מודל מסוגנן ומופשט של הארגון

מרכיב המידע משמש כמתווך בין מרכיב התפעול למרכיב הניהול ונותן למרכיב הניהול תמונה מופשטת על המתרחש במרכיב התפעול. קיימת אבחנה ברורה בין הזרימה הפיסית המתרחשת במרכיב התפעול לבין זרימת המידע במערכת המידע.

ניתוח מערכת הארגון

ניתוח מערכת הארגון פירושו ביצוע סקר שיטתי שתוצאותיו בניית תמונה מפורטת דיה של הארגון שאותו מנתחים, באמצעותה ניתן לאתר את צרכי המידע שלו, את נמעני המידע, את מיקומה של מערכת המידע בארגון וכן את מדיניות הקמתה של מערכת המידע.

ביצוע הסקר פירושו חיפוש תשובות לשאלות הבאות:

א. מהן מטרותיו, יעדיו ומדדיו של הארגון.
ב. מהם גבולות הארגון.
ג. מהם משאבי הארגון
— זרמי המשאבים הנכנסים אליו
— זרמי המשאבים המצויים בו.
ד. מה הגורמים הבונים את הארגון ומהם התהליכים המתבצעים בהם. את התהליכים יש לסווג לתהליכים תפעוליים, ניהוליים ושל מידע.
ה. מהם קשרי הגומלין בין הגורמים.
ו. מהי התייחסותו של כל גורם במערכת למטרות הארגון.
ז. מהי הסביבה הרלוואנטית של הארגון (הקרובה והרחוקה), מהם קשרי הגומלין של המערכת עמה וכן מהם האילוצים והמגבלות שסביבה זו מפעילה על המערכת.
ח. מהם דרכי הפעולה העומדות לרשות הארגון המאפשרות לו להשיג את מטרותיו וכן מהי יכולתו ליצור דרכי פעולה חדשות בעת הצורך.

מערכת הניהול

מרכיב הניהול בארגון הנו תת־המערכת של הארגון שהוא הצרכן העיקרי של מערכת המידע, זאת, משום היות המידע המופק על ידה כלי הניהול העיקרי. לפיכך, אין בנייתה של מערכת המידע יכולה להתבצע נכונה אם לא נבין את מהותה וצרכיה של מערכת הניהול שאותה היא משרתת. תהליך הניהול הנו תהליך של טרנספורמציה שבו הופך מידע (הקלט) להחלטות (הפלט). מטרתו של תהליך הבקרה על המרכיב התפעולי ומשמעותה של הבקרה הנה אחריות להפיכתה של מערכת משאבים — המרכיב התפעולי — (כוח אדם, חומרים, טכנולוגיות, אנרגיה, הון, אמצעים פיסיים ומידע) למערכת הרמונית (שילוב, איחוד וקישור בין מרכיביה של מערכת המשאבים) המסוגלת להשיג ביעילות (מוכתבת על־ידי מדדי הביצוע) את המטרות שהוצבו בפניה (הפקת הפלט שהוא: מוצרים, שירותים או רעיונות).

תכונות היסוד של תהליך הניהול — עובדת היותו בעיקרון אותו תהליך עצמו בכל דרגי הניהול של הארגון, ממנהל העבודה ועד המנכ״ל. התהליך הנו אמנות עם בסיס מדעי ומשום היותו כזה תלותו באדם המבצע אותו מכרעת.

הפונקציות המרכיבות את תהליך הניהול

דרך הניתוח המתאימה לצרכינו היא דרך פירוקו של תהליך הניהול לפונקציות שממנו הוא מורכב. אין מדובר כאן בפונקציות מקצועיות כמו: שיווק, ייצור, כספים, כוח אדם וכו׳, אלא, בפונקציות הניהול האוניברסליות אשר מבצעים מנהלים בכל הדרגים.

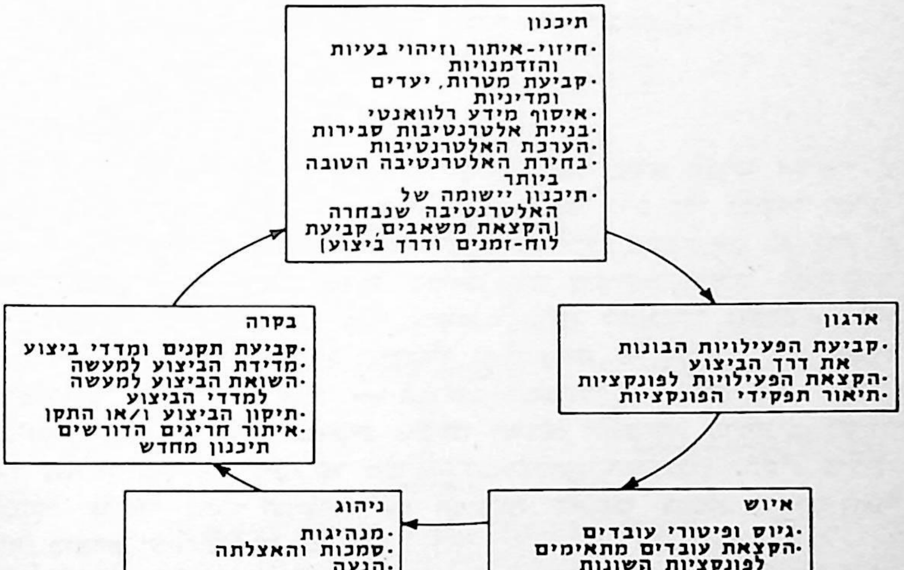
משקלן הסגולי של הפונקציות משתנה מדרג ניהול אחד למשנהו (מבחינת הדגשים, היקף, חשיבות, כמות הזמן המושקעת וכו׳).

הפונקציות שמהן בנוי תהליך הניהול הן: (תרשים 3.3).

— תכנון (planning)
— ארגון (organizing)
— איוש (staffing)
— ניהוג (directing)
— בקרה (controlling)

עם זאת, קיימות פעילויות ניהוליות הבאות לידי ביטוי בכל אחת מפונקציות הניהול והן:

— קבלת החלטות
— פעילות-גומלין אנושית
— תיאום ותקשורת
— מיבנות (structuring) פעילויות.



תרשים 3.3: תהליך הניהול

חלוקת תהליך הניהול לפונקציות המרכיבות אותו היא פעולה מתודולוגית במהותה, לאמור: פירוקו של התהליך לחמשת הפונקציות הללו דווקא הוא מעשה שרירותי במידה רבה ותכליתו לימוד וניתוח תהליך הניהול. הגבולות בין הפונקציות אינם חד־משמעיים וביצועו של התהליך במציאות אינו מתבצע דווקא בסדר מוכתב מראש, תוך הפרדה בין הפונקציות השונות.

כאמור, נועדה הפרדת התהליך לפונקציות ללימוד וניתוח שמטרתם איתור צרכי המידע של מערכת הניהול המבצעת את תהליך הניהול. ואמנם כל פונקציה ניהולית וצרכיה היחודיים, כל פונקציה ויכולת שונה של מערכת המידע הממוחשבת לתמוך בביצועה. אם נדרג את הפונקציות בהתאם ליכולת מערכת המידע הממוחשבת לתמוך בהן נקבל את הדירוג האיכותי הבא:

— פונקצית הבקרה — תמיכה רבה
— פונקצית התכנון — תמיכה בינונית
— פונקציות האיוש, הארגון והניהוג — תמיכה מועטת.

קבלת החלטות (תרשים 3.4)

הפעילות הניהולית הבאה לידי ביטוי בכל פונקציות הניהול ואשר מאפיינת יותר מכל את תהליך הניהול כולו היא פעילות קבלת ההחלטות.

כאמור, אחד מהגדרות תהליך הניהול קובעת שתהליך זה הנו ביצוע טרנספורמציה של מידע להחלטות.

היכולת לאתר את ההחלטות הנעשות בארגון וניתוח תהליך עשייתן הנם בסיס האיתור של צרכי המידע לניהול של הארגון.

את המידע בכללותו ניתן לחלק לשתי קטגוריות:
— מידע לתפעול.
— מידע לניהול.

המידע לתפעול, מצידו, מתחלק לשני סוגים:
— מידע פקידותי — מערכת המידע הממוחשבת מחליפה עבודת ידיים בעבודת מכונה. לדוגמא: הפקת תלוש השכר, הפקדת הזמנה לספק וכו׳.
— מידע של פעולה — מערכת המידע הממוחשבת היא שמקבלת את ההחלטה. לדוגמא: בחירת ספק מסוים, קביעת לוח־זמנים לייצור.

מידע לניהול הנו מידע התומך בתהליך קבלת ההחלטה כשהתהליך עצמו נתון בידיו ובאחריותו של המנהל והמידע משמש לו ככלי העשוי לשפר את טיב התהליך.

ככל שתהליך קבלת ההחלטה מוגדר טוב יותר כן רבה יותר יכולתה של מערכת המידע הממוחשבת לתמוך בתהליך.

תהליך קבלת החלטה אינו פעולה בנקודת זמן. מספר שלבים מרכיבים אותו שלכל אחד תפקידו הייחודי והתייחסות מוגדרת למערכת המידע הממוחשבת.

לתהליך שלושה שלבים עיקריים המתפצלים למספר תת־שלבים.

שלושת השלבים העיקריים הם:

— שלב ההכרה (cognition)
— שלב הניתוח והתיכון (analysis and design)
— שלב הבחירה (choice)

כל שלב עושה שימוש במידע. עקרונית, לא חייב אותו גורם לבצע את התהליך כולו בשלמותו וניתן להפקיד כל חלק מהתהליך בידי גורמים שונים (מחשב, פונקציות מטה, המנהל המחליט).

ביצוע התהליך בצורה נכונה פירושו ביצוע כל שלביו בסדר המוכתב. אולם, יש לזכור, שהתהליך הנו איטרטיבי במהותו.

כדי לעשות שימוש נכון בתהליך קבלת ההחלטה יש להכיר את פרטיו שהם:

א. שלב ההכרה

— איתור וזיהוי הבעיה או ההזדמנות (תפישה)
— הגדרת הבעיה או ההזדמנות (ניסוח)
— קבלת החלטה שאמנם יש מקום להמשיך בתהליך (שאמנם קיימת בעיה שיש ואפשר לפותרה).

ב. שלב הניתוח והתיכון

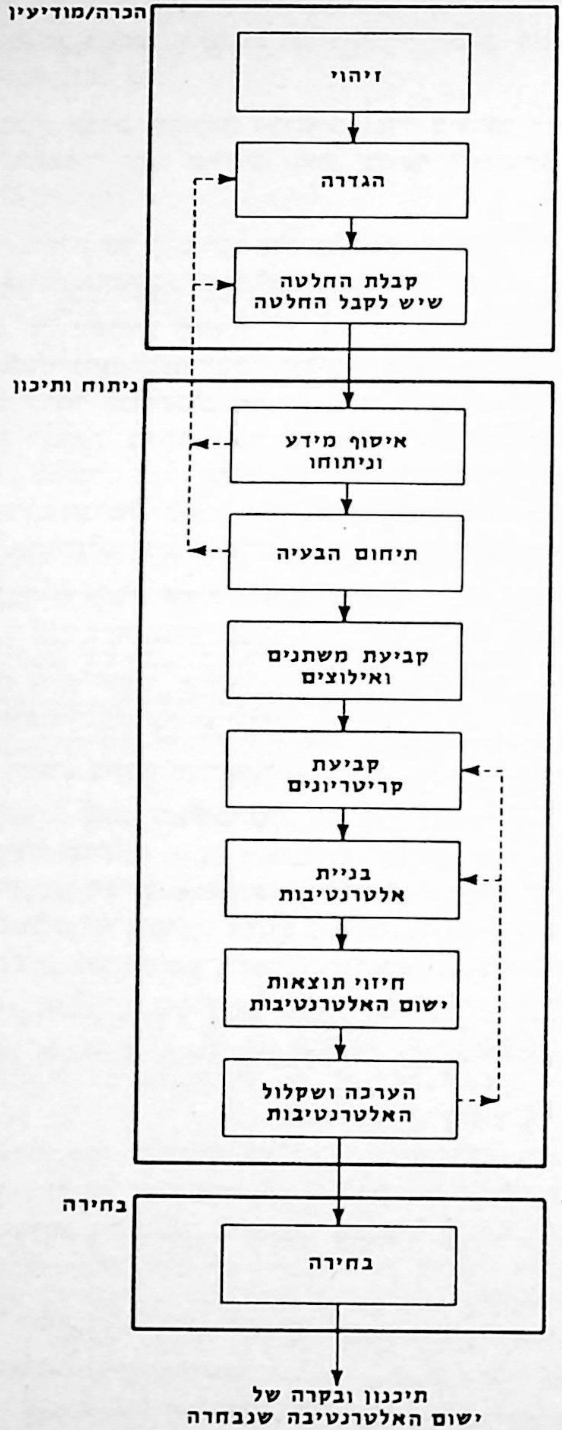
— איסוף מידע (על הסביבה ועל המערכת שבתוכה מקבלים את ההחלטה) וניתוחו;
— איתור וקביעת המשתנים והאילוצים הרלוואנטים לבעיה או להזדמנות;
— הגדרת הקריטריונים לפיהם ישוקללו האלטרנטיבות שייבחנו;
— בניית אלטרנטיבות סבירות, לאמור: תיכון דרכי פעולה שביצוען יביא לפתרון הבעיה או לניצול ההזדמנות;
— חיזוי תוצאות יישום האלטרנטיבות (על־ידי ניסוי, מודל וכו׳);
— השוואת התוצאות החזויות לקריטריוני הבחירה ושקלול האלטרנטיבות בהתאם.

ג. בחירה

— בחירת האלטרנטיבה הטובה ביותר.

אין תהליך קבלת ההחלטה מסתיים בעצם הבחירה. קיימים עוד שני שלבים, שהם למעשה חלק בלתי נפרד מהתהליך (הנקראים post-decision phases) והם:

— תכנון התקנת האלטרנטיבה שנבחרה
— בקרה על תוצאות ביצוע האלטרנטיבה שנבחרה.



תרשים 3.4 : תהליך קבלת החלטה

בנקודה זו יש מקום למספר הבהרות על תהליך קבלת ההחלטה וחלקה של מערכת המידע הממוחשבת בו:

א. כל שלבי תהליך קבלת ההחלטה הכרחיים וסדר ביצועם הכרחי אף הוא (אם לא מופסק התהליך לפני סיומו). זאת, בנוסף לאפשרות שהתהליך יהיה איטרטיבי.

ב. קיימים קשרי גומלין של קלט ושל פלט בין שלבי התהליך.

ג. מערכת המידע הממוחשבת מסוגלת לטפל אך ורק בהחלטות המוגדרות כרציונאליות. רציונאליות בהקשר זה פירושה דבקות בקריטריונים לבחירה, לאמור: דבקות בהם בעת הבחירה בין האלטרנטיבות. תחילה נקבעים הקריטריונים, לאחר מכן האלטרנטיבות וחיזוי תוצאות יישומן ולבסוף בחירת האלטרנטיבה הטובה ביותר לאור ההשוואה בין התוצאות החזויות לבין הקריטריונים. במסגרת של קבלת החלטות רציונאליות, הבחירה החופשית קיימת רק בשלב קביעת הקריטריונים ולא בשלב הבחירה. זאת, כמובן בתנאי שניתן לסדר את האלטרנטיבות בסדר התואם את הקריטריונים.

ד. מידע דרוש לתהליך קבלת ההחלטה בכל שלביו.

ה. חיזוי תוצאות היישום של האלטרנטיבות הסבירות יכול להעשות על־ידי ניסוי למעשה (דרך יקרה, ארוכה ומורכבת) או על־ידי בניית מודל האמור לתאר את התנהגות המערכת הריאלית.

ו. בשלב ניתוח המידע מתבצעות הפעולות הבאות:
— בדיקת עקביות הנתונים (אימות)
— סידור המידע (מיון)
— אבחנה בין טפל לעיקר — סלקציה (שליפה)
— ביצוע חישובים (חישוב)
— הצגת תוצאות (הצגה).

פעולות מבוצעות על־ידי מערכת מידע ממוחשבת מצוינות בסוגריים, אם אמנם שלב זה נמסר לה. מסירה זו יכולה להתבצע רק אם דרך הניתוח (האלגוריתם) תוגדר מראש.

סוגי החלטות

יכולתה של מערכת מידע ממוחשבת לתמוך בתהליך קבלת ההחלטה תלוי בסוג ההחלטה.

את אוכלוסיית ההחלטות ניתן לחלק לשני סוגים:
— החלטות מובנות (structured)
— החלטות שאינן מובנות (unstructured)

החלטה מובנית, היא החלטה שתהליך עשייתה הוא אלגוריתמי (algorithm), לאמור:

* לוגי
* כמותי
* חד־משמעי
* מוגדר לחלוטין.

החלטה זו מתקבלת בתנאי ודאות — לאמור, כל האלטרנטיבות ותוצאות יישומן ידועות וכמותיות וניתן להשוותן מול קריטריון כמותי.

החלטה בלתי־מובנית, היא החלטה שתהליך עשייתה הוא היאוריסטי (heuristic), לאמור:

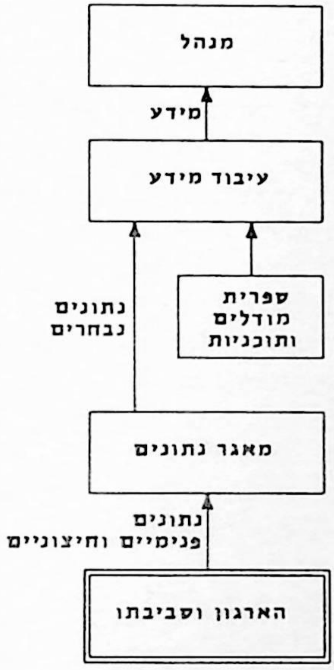
* אינו מוגדר לחלוטין
* לא ניתן לאתר ולהגדיר את כל המשתנים
* לא ניתן לכמת את כל המשתנים המוגדרים.

כדי לקבל החלטות בלתי־מובנות יש לשער השערות, לעשות שימוש באינטואיציה, הערכה, שיקול דעת, נסיון ולזכות במזל. זו ההחלטה המתקבלת בתנאי אי־ודאות — אין ודאות שדרך הפעולה שעליה יוחלט היא הטובה ביותר ושאמנם תשיג את המטרה שאליה חותרים. זאת משום שלא כל האלטרנטיבות ידועות ומשום שלא ניתן לחזות את תוצאות היישום של האלטרנטיבות הידועות.
אוכלוסיית ההחלטות הלא־מובנות היא האוכלוסייה הגדולה, אולם ניתן להבחין בתוכה בין החלטות מובנות יותר ומובנות פחות.
תכונת המיבניות היא תכונת יסוד של כל החלטה. החלטות מובנות הנן ברות־תיכנות (מתוכנתות) (programmed decisions) ואילו החלטות שאינן־מובנות אינן ברות תיכנות (unprogrammed decisions) כשהמושג תיכנות בהקשר זה מובנו פיתוח תהליך קבלת החלטות מוגדר, מפורט, שלם וחד־משמעי, לאמור: הכנסת תהליך קבלת ההחלטה למסגרת מוכתבת וקבועה מראש.
ההחלטות המתוכנתות הנן בדרך כלל ההחלטות החוזרות, הרוטיניות ואילו ההחלטות הבלתי־מתוכנתות הן החלטות חדשות, חד־פעמיות. עם זאת, לא כדאי לתכנת כל החלטה מובנית. ההחלטה אם לתכנת החלטה מובנית או לא, חייבת להיות פועל יוצא של ניתוח עלות/תועלת.
יתירה מזו: החלטות בלתי־מובנות, גם אם בתחילת קיומן אינן כאלו הרי אם הן חוזרות על עצמן קיימת הסתברות גבוהה, שעם הזמן, יהפכו לכאלו.
רק החלטות מובנות שתהליך קבלתן רציונאלי הנן ברות מיחשוב. החלטות כאלו ניתן למסור למערכת המידע הממוחשבת. כל שאר סוגי ההחלטות אינן ברות מיחשוב ובתהליך קבלתן חייב האדם להיות מעורב. מערכת המידע הממוחשבת יכולה לתרום על־ידי הפקת מידע לניהול התומך בהתהליך קבלת ההחלטה, אולם, אינה מסוגלת לעשותו באופן בלעדי. תמיכתה של מערכת המידע הממוחשבת יכולה לבוא לידי ביטוי בכל אחד משלבי תהליך קבלת ההחלטות.

לסיכום ניתן לאמור: מערכת המידע ממוחשבת מסוגלת לקבל החלטה אם יתמלאו שני תנאים:

— תהליך קבלת ההחלטה יוגדר לחלוטין ויתוכנת
— המידע הדרוש לתהליך יימצא במערכת ברמת שלמות ואמינות התואמת את צורכי התהליך.

לעומת ההחלטות המובנות, מערכת המידע הממוחשבת יכולה רק לתמוך בתהליך קבלת החלטה שאינה מובנית. ככל שההחלטה מובנית יותר כן גדולה יותר יכולת התמיכה של מערכת המידע.



תרשים 3.5: תהליך קבלת החלטה בלתי מתוכנתת בתמיכת מערכת מידע ממוחשבת

מאפייני ההחלטות ביחס לדרג הניהול המקבל אותן

מקובל לאבחן שלושה דרגי ניהול במערכת הניהול:

— הדרג הסטרטגי (strategic planning)
— הדרג הטקטי (management control)
— הדרג התפעולי (operational control).

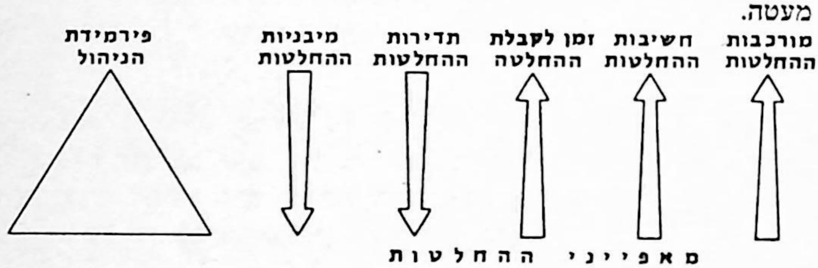
עקרונית, כל דרג ניהול מקבל החלטות בעלות מאפיינים ייחודיים לאותו דרג. נאמר "עקרונית" משום שבמציאות אין החלוקה ברורה וחד־משמעית.

לצורך עניינו ניתן לאמור שקיימים שלושה סוגי החלטות:
— החלטות סטרטגיות
— החלטות טקטיות
— החלטות תפעוליות.

לכל סוג החלטה מאפייני המידע התומך בו המכתיבים את המיבנה של מערכת המידע האמורה לתמוך בהחלטות אלו.

הפעילויות הניהוליות שמבצע כל דרג ניהולי כמפורט מעלה מפורטות להלן:
א. הדרג הסטרטגי: זה עוסק בהצבת מטרות הארגון ובעריכת שינויים בהן, בהחלטה על המשאבים הדרושים להשגת המטרות ובמדיניות לפיה יעשו בהם שימוש, לאמור: הדרג הסטרטגי עיסוקו בעיקר בתכנון לטווח ארוך.
ב. הדרג הטקטי עוסק בקביעת יעדים, שהם תרגום המטרות שקבע הדרג הסטרטגי, בתכנון מפורט של אופן השגתם ובבקרה על ביצוע התכנון, לאמור: תכנון לטווח קצר עד בינוני במסגרת מטרות כלליות נתונות.
ג. הדרג התפעולי עיסוקו בניהול יומיומי של הביצוע במסגרת תהליכי העבודה, התכנון והיעדים שקבע הדרג הטקטי.

ניתוח פעילויות אלו מורה כי רבה יכולתה של מערכת המידע הממוחשבת לתרום לתהליכי הניהול התפעוליים, ואילו לתהליכי הניהול הסטרטגיים תרומתה האפ־שרית מעטה.



תרשים 3.6: איפיון איכותי של ההחלטות ביחס לדרג הניהול שבו הן נעשות.
(ראש החץ מבטא "הרבה", זנבו — "מעט")

תכונות מערכת המידע בהתייחס לסוג ההחלטות שהיא תומכת בהן.
התכונות העיקריות המעצבות את מערכת המידע הממוחשבת הן:
— מקור הנתונים — האם המקור מצוי בתוך מערכת הארגון או מחוצה לה;
— מידת העדכניות של הנתונים האגורים במערכת. עדכניות פירושה: פער הזמן בין מועד התרחשות האירוע לבין מועד הזמינות של המידע על האירוע;
— אורך חייהם של הנתונים, לאמור: כמה זמן לאגרם;
— מידת התמצות, לאמור: באיזו צורה לאגור את הנתונים;
— מידת ההיקף, לאמור: באיזו מידה מקיפה המערכת את הארגון כולו או את חלקו בלבד;

— מהירות הגישה. מהירות גישה פירושה: פער הזמן העובר בין מועד הדרישה למידע לבין מועד קבלתו;
— צורת הגישה, לאמור: דו"ח ניר, צג, טלקס וכו'.

הטבלה הבאה מציגה את הקשר שבין תכונות המידע וסוגי ההחלטות בארגון:

מאפייני המידע / סוג ההחלטה	החלטה סטרטגית	החלטה טקטית	החלטה תפעולית
עדכניות	בינונית	גבוהה	גבוהה מאד
אורך חיים	ארוך	בינוני	קצר
תמצות	גבוה מאד	בינוני	אין תמצות
היקף	רחב מאד	מוגבל	צר
מהירות גישה	איטית	בינונית	מהירה
צורת גישה	— שאילתות — דוחות אד־הוק — דו־שיח אדם־מחשב	— דוחות שוטפים — דוחות חריגים — שאילתות	גישה ישירה (on-line)
מקור הנתונים	בעיקר חיצוני	בעיקר פנימי	פנימי

למידע המופק ממערכת המידע יש ערך רק כאשר עושים בו שימוש. במידע ייעשה שימוש רק אם טיבו יתאים לצרכים שלשמם הוא מופק. לפיכך, חובה לאתר את צרכי המידע טרם בנייתה של מערכת המידע ולבחון את המידע האמור להיות מופק על ידה לפי המאבחנים האיכותיים הבאים:

— רלוואנטיות: האם קיים קשר בין המידע המופק למטרה שלמענה הופק?
— צורניות: האם הצד הצורני של המסמך נושא המידע הוא כזה שהאדם שיקבלו יהיה מסוגל להבין את תוכנו?
— עיתוי: האם המידע מופק בזמן הנכון?
— יעד: האם המידע מופנה לבעל־התפקיד הנכון?
— תמצות: האם מידת התצמות תואמת את הצרכים?
— כדאיות: האם ערך המידע המופק שווה את עלותו?
— שלמות: האם המידע מקיף את הצרכים שלשמם הופק?
— אמינות: האם דיוקו של המידע המופק תואם את הצרכים?

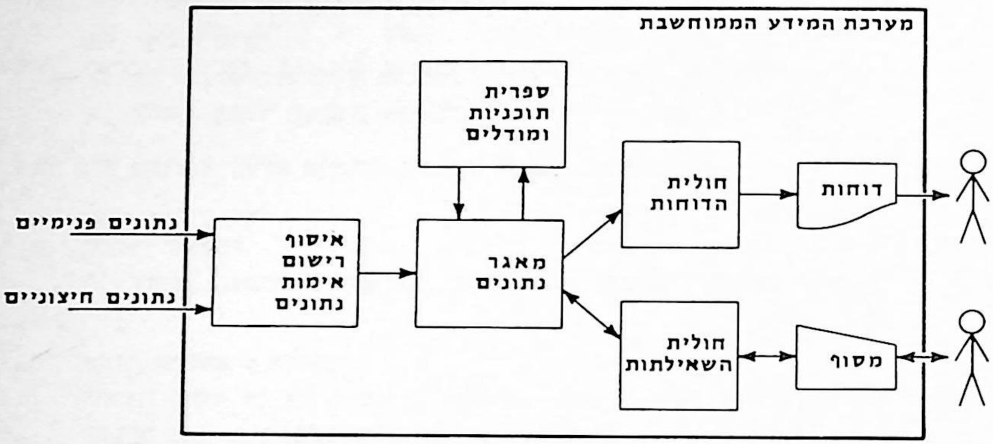
המטרה היא שההתאמה בין הצרכים לבין המידע המופק ממערכת המידע תהיה הטובה ביותר ובה בשעה לשמור על האיזון בין העלות לתועלת.

מבנה המערכת

את מבנה מערכת המידע הממוחשבת נתאר משני היבטים שונים:

— היבט המבנה התפקודי.

— היבט המבנה הפיסי.



תרשים 3.7: **מבנה תיפקודי** של מערכת מידע ממוחשבת

א. היבט המבנה התפקודי

מערכת המידע הממוחשבת בנויה מחמש חוליות (מודולים) תפקודיות לוגיות:

1. החוליה האוספת, רושמת ומאמתת את נתוני הקלט (data gathering module).
2. החוליה האוגרת את הנתונים (data base).
3. החוליה שבה מצויות התוכניות והמודלים המופעלים על הנתונים להפקת מידע (model and program base).
4. החוליה המפיקה דוחות שוטפים וחריגים (מידע פאסיבי, report module).
5. החוליה שבאמצעותה ניתן לקבל תשובות לשאילתות ובאמצעותה יכול המנהל לשוחח עם מערכת המידע (query module).

ב. היבט המבנה הפיסי

מערכת המידע הממוחשבת בנויה מארבעה מרכיבים פיסיים:

1. חומרת (hardware) מחשב (יחידת עיבוד מרכזית, זיכרון, אמצעי אחסנה, אמצעי תקשורת, ציוד היקפי וכו׳).
2. תוכנת מחשב (מערכת הפעלה, תוכנת תקשורת, תוכנת מסדי נתונים, שפת שאילתות, תוכניות וכו׳).

3. נהלים ידניים הקובעים את דרכה ופעולתה של המערכת (מי מזרים, מה ומתי אל המחשב ומי מקבל מה ומתי מהמחשב). ניתן להבחין בנהלי משתמש, נהלי קליטה ואימות ונהלי הפעלה.
4. כוח־אדם: מנתחי מערכות מידע, תוכניתנים, מפעילים.

האפשרויות לאיסוף הנתונים בארגון הן:
א. איסוף מרוכז;
ב. כל פונקציה ארגונית מבצעת את האיסוף שהוא רלוואנטי לה;
ג. איסוף מבוזר המשרת את כל הפונקציות.

את פלט מערכת המידע הממוחשבת ניתן לסווג בצורה הבאה:

א. מסמכי תנועות
כמו למשל: חשבונות ללקוחות, תלושי שכר לעובדים, הזמנות לספקים.

ב. דוחות מתוכננים מראש
התכנון מקיף הן את תוכנו וצורתו של הדוח והן את קצב הפקתו כשהמניע להפקה הוא הזמן (פעם ביום, פעם בשבוע וכו׳).
הדוחות הללו מתארים את שהתרחש בתקופה מסוימת או את המצב הקיים בנקודת זמן מסויימת.

ג. דוחות חריגים
דוחות המופקים עם תחילתו של חריג, כפי שהוא מוגדר על־ידי התוכנית. המניע להפקתם של דוחות אלה הוא האירוע ולא הזמן כפי שהדבר קיים בדוחות מתוכננים מראש.

ד. דוחות חד־פעמיים ותשובות לשאילתות
שאילתות יכולות להיות מתוכננות מראש ולא מתוכננות מראש. השאילתות מכוונות למידע ספציפי שאינו מצוי במסגרת הדוחות הרגילים (תקופתיים או חריגים) או שהצורך בו אינו בר־חיזוי מראש, כמו למשל: מצבו של פריט מסוים.
תשובה לשאילתא יכולה להינתן ב־on-line או ב־off-line, בתלות במבנה מערכת המידע ומערכת המחשב שבה עושים שימוש.
דוחות חד־פעמיים באים לענות על צורך שמתכני מערכת המידע לא צפו מראש (הדוח החד־פעמי יכול ליהפך לדוח שגרתי). במקרה זה יש להבחין בין מידע הניתן להפקה מתוך מאגר הנתונים המצוי במערכת לבין מידע שלשם הפקתו יש צורך באיסוף נתונים מיוחד (או שילוב של שני מקרים אלה).

בניית תוכנית השאילתא או בניית התוכנית המפיקה את הדוח החד־פעמי יכולה להעשות או על־ידי היחידה המקצועית או על־ידי המשתמש עצמו.

ה. דו־שיח אדם/מחשב

זהו מקרה שבו המשתמש יכול להגיע בעזרת אינטרקציה עם ספריות המודלים, התוכניות ומאגר הנתונים, לפתרון מספק של בעיה. דו־שיח אדם/מחשב מכתיב שימוש במסוף וקיומן של ספריות תוכניות, מודלים ומאגר נתונים ברי־גישה המכילים את הדרוש לבעל הבעיה.

2.

מחזור החיים של מערכת מידע ממוחשבת

מבוא

ניתן להגדיר את מושג מחזור החיים כתהליך הפיתוח וההפעלה של מערכות מידע ממוחשבות. נושא מחזור החיים של מערכת מידע ממוחשבת מקיף את הנושאים הבאים:

א. מאפייני הסביבה שבה מתבצע תהליך הפיתוח וההפעלה.
ב. עקרונות יסוד של תהליך הפיתוח וההפעלה.
ג. תהליך הפיתוח וההפעלה — מרכיביו ומבצעיו.
ד. ארגון עבודת הפיתוח.

הגישה לתהליך הפיתוח וההפעלה של מערכות מידע ממוחשבות מבטאת את הצורך שתהליך יהיה מתוכנן ומבוקר.
מערכת מידע ממוחשבת מורכבת ממספר רב של מרכיבים המקיימים ביניהם יחסי גומלין. בתהליך הפיתוח וההפעלה משתתפים גורמים רבים ושונים מכל חלקי הארגון שהם בעלי רקע מקצועי, ניהולי ומעמדי שונה. מצב עניינים זה, המבטא את המורכבות וההיקף של הפיתוח וההפעלה של מערכות מידע, מכתיב את הצורך שהתהליך יוכנס למסגרת מוגדרת ובת־שליטה. כך ניתן יהיה לחלק את מערכת המידע לתת־מערכות תוך הגדרת המימשקים ביניהן, ולטפל במרכיבים ברי־שליטה וברי־פיתוח תוך ידיעה, שבסופו של התהליך יתחברו המרכיבים למערכת הרמונית שתשיג את מטרותיה ביעילות מירבית.
רעיון מחזור החיים אינו מוגבל למערכות מידע בלבד. קונספקציה אנאלוגית מיושמת, לדוגמא, בפיתוחו של מוצר חדש. ההבדל בין מערכות מידע למוצר כלשהו הוא בכך, שמוצר הוא עצם פיסי־מוחשי ואילו מערכות מידע הנן עצם מופשט. הרעיון המרכזי הטמון ביסוד מחזור החיים הוא, שפיתוחה והפעלתה של

כל מערכת חייבים לעבור דרך אותו תהליך שיטתי ולוגי המורכב משלבים מוגדרים שעליהם לא ניתן לפסוח ושאת ביצועם אין לשנות. בנייתה של מערכת מידע ממוחשבת דומה לבניית בית, שהוא מוצר חד־פעמי. ברוב המקרים, אין דומה בית אחד למשנהו, אולם הבנייה זהה.

להלן חלק מהבעיות שבהן מתלבטים ארגונים המפתחים מערכות מידע ממוחשבות:
— ההשקעות גדולות מהמתוכנן הן בפיתוח והן בהפעלה.
— לוח הזמנים לפיתוח ארוך מדי ומכאן שזמן התגובה ארוך אף הוא.
— אין המפתחים עומדים בלוח הזמנים המתוכננים.
— דרישות עיקריות למידע אינן נכללות במערכות שמפתחים.
— התקנת המערכות בארגון כרוכה במאמצים רבים ובתקלות.
— הפלט, גם לאחר תום הפיתוח, אינו אמין.
— התועלת המתוכננת אינה מושגת.

הסיבות העיקריות לבעיות אלו הן:
— היקף המערכות רחב מדי.
— בעת הפיתוח, מצמצמים את היקפו הרחב, לפיכך תועלת המערכת אינה זו המובטחת.
— אין מקדישים זמן ניכר לשלב ההגדרה. מטעם זה תוצאותיו הן מיפרטים בלתי־שלמים.
— אי־קיום מסגרת שיטתית שלמה וברורה לבקרה של תהליך הפיתוח.

הצעדים שיש לנקוט כדי למנוע בעיות אלו:
— מיבנות תהליך הפיתוח של מערכות מידע ממוחשבות.
— קביעת נקודות בקרה והחלטה מוגדרות ונוהל ברור לפעילות בנקודות אלו.

מאפייני הסביבה שבה מתבצע תהליך הפיתוח וההפעלה של מערכות מידע ממוחשבות

בפיתוח מערכת מידע חייבים להילקח בחשבון מספר מאפיינים שהשפעתם על תהליך הפיתוח מכרעת:

א. התפתחויות מהירות ומשמעותיות בשטח הציוד והתוכנה. השפעת התפתחויות אלה על כדאיות או אי־כדאיות מיחשובן של מערכות מידע הוא רב.

ב. זמן הפיתוח של המערכות ארוך. זאת, משום שהמערכות של היום הנן מורכבות ובעלות היקף רחב (הן משום שזהו הצורך והן משום שניתן לפתח מערכות כאלו).

ג. ההחלטה לפתח מערכת מידע ממוחשבת בשלב מוקדם יחסית של תהליך הפיתוח. מקבלי ההחלטה אינם יכולים בשלב זה לדעת אל נכון את מהותה של המערכת שהחליטו להשקיע בה משאבים רבים יחסית.

ד. היקף המשאבים הדרושים לפיתוח ולהפעלה של מערכות המידע הממוחשבות הולך וגדל עם הזמן.

ה. ההיצע של כוח האדם המקצועי האמור לעסוק בפיתוח ובהפעלה של מערכות המידע הממוחשבות הן מבחינת איכותו והן מבחינת כמותו אינו מדביק את הביקוש.

ו. קיימים קשיי תקשורת עקרוניים בין העוסקים בפיתוח לבין המשתמשים במערכות (הלקוחות). עובדה זו באה לידי ביטוי הן בשלב איתור צרכי המידע, הן במהלך תהליך הפיתוח והן בעת ההפעלה של המערכות.

עקרונות יסוד של תהליך הפיתוח וההפעלה של מערכות מידע ממוחשבות

א. ההשקעה בפיתוח מערכות מידע ממוחשבות הנה השקעה לכל דבר ויש לטפל בה כפי שמטפלים בכל השקעת הון, כלומר, להצדיק את ההשקעה.

הגורמים הבאים צריכים, אפוא, להילקח בחשבון:

— הערך הנוכחי של המערכת לאור ההשקעות בה בהווה ובעתיד (הוצאות הפיתוח ועלות ההפעלה).
— אורך החיים המתוכנן של המערכת.
— מחיר הגרט של המערכת הישנה המוחלפת בחדשה שאותה מפתחים.
— בסביבה של משאבים מוגבלים, יש לבדוק אם המערכת שעומדים לפתח היא בעלת הערך הגבוה ביותר מבין המערכות שניתן לפתח.

ב. לכל מערכת מידע חייבים להיות מדדים כמותיים שיאפשרו מדידת הצלחתה של המערכת, כלומר באיזו מידה השיגה את מטרותיה. רצוי, שהמערכת עצמה תפיק מידע שיאפשר את מדידת הצלחתה.

ג. המערכת שאותה מפתחים נועדה לשרת את המשתמש. פירושה של קביעה זו הוא שדרושה מעורבותו הפעילה של המשתמש בתהליך הפיתוח של מערכת המידע. המפרטים של המערכת והשינויים במפרטים אלה חייבים לקבל את אישורו של המשתמש לפני המעבר לשלב הבנייה. המצב הרצוי הוא שיתופו הפעיל של המשתמש בבניית המפרטים. המשתמש חייב ליטול על עצמו את האחריות להתקנתה של המערכת בארגונו ולאשר את נהלי השימוש במערכת לפני הפצתם. המצב הרצוי במקרה זה הוא שיתופו של המשתמש בכתיבת הנהלים.

ד. מאחר ותהליך הפיתוח של מערכת מידע ממוחשבת הנו ארוך ומורכב, יש לחלקו לקטעים, שבסופם נקודות בקרה (אבני־דרך) שבהן מתבצעת בקרה על העבודה שנעשתה עד לנקודת הבקרה, וכן בקרה על תכנון העבודה שיש עוד לבצע עד לסיום העבודה.

תהליך הפיתוח וההפעלה של מערכות מידע מתחלק לארבעה שלבים עיקריים (המתחלקים אף הם לתת־שלבים, עליהם ידובר להלן):

1. שלב ההגדרה.
2. שלב הבנייה.
3. שלב ההתקנה.
4. שלב התפעול והאחזקה.

שלושת השלבים הראשונים מתייחסים לתהליך הפיתוח, השלב הרביעי לתהליך ההפעלה. בתהליך הבקרה טמונים מספר קשיים: מבקר מן החוץ אינו "חי" את המערכת טרם תפעל; המפתחים מגינים ב"חום" על עבודתם; בדרך כלל נמצא תהליך הפיתוח בלחץ זמן; קיימת נטייה להקצות לעבודת הביקורת עובדים בדרג נמוך, שאינם מסוגלים לעמוד בלחץ מפתחי המערכת.

ה. אין להסס מלהפסיק פיתוחה של מערכת. השלב שבו רצוי לקבל החלטה כזו הוא שלב ההגדרה. זאת, משום שהמשאבים המבוזבזים בגין ההפסקה עדיין אינם רבים. לשון אחר, בפיתוחה של מערכת מידע ממוחשבת כרוכים סיכונים, שהעיקריים הם:

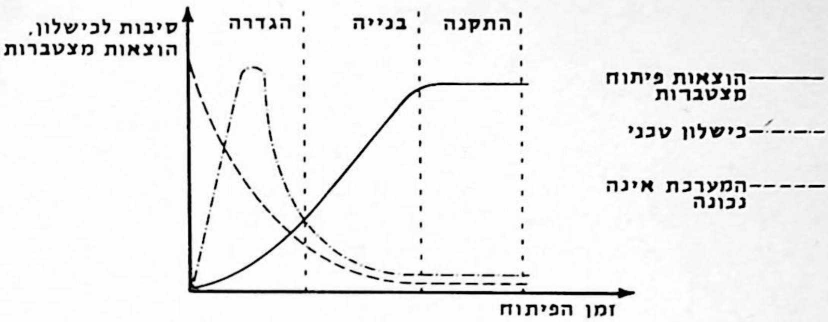
— המערכת אינה זו הנכונה (אינה עונה על בעיות הארגון או אין היא מסוגלת לפתור את בעיות הארגון).
— המערכת אינה ישימה מהבחינה הטכנית.
— המערכת אינה ישימה מהבחינה התפעולית.

שלב ההגדרה הוא השלב שבו יש להסיר סיכונים אלה. לפיכך, יש להקצות לשלב ההגדרה, שהוא הפחות ממוסד ושבו מומנט היצירתיות בא לידי ביטוי, את אנשי המקצוע הטובים ביותר. אין ללחוץ על קיצורו של שלב זה, אלא להקצות לו זמן שיאפשר הסרת הסיכונים, מה עוד ששלב זה הנו יחסית הזול ביותר בשלבי הפיתוח של המערכת.

תרשימים 3.8 — 3.9 ממחישים את האמור לעיל:



תרשים 3.8: חלוקת ההשקעה בכוח אדם בין שלבי הפיתוח השונים



תרשים 3.9: עקומות סיבות לכישלון בפיתוח מול עקומת הוצאות פיתוח מצטברות.

מודגש, אפוא, שאין לחסוך בזמן ובהוצאות בשלב ההגדרה, כדי למנוע סיכונים בהמשך הפיתוח. רצוי להפסיק פיתוחה של מערכת, אם יש צורך, כשתהליך הפיתוח נמצא עדיין בשלב ההגדרה. שלב הגדרה גרוע עלול לגרום לבעיות בשלבים הבאים (כגון שינויים במפרטי התיכנות בשלב התיכנות).

ו. המעורבות האקטיבית של הנהלת הארגון בתהליך הפיתוח חייבת לבוא לידי ביטוי בעיקר בשלב ההגדרה, עקב חשיבותו של שלב זה. ביצוע נכון של שלב ההגדרה מבטיח במידה רבה את הצלחתם של השלבים הבאים אחריו. נוסף על שלב ההגדרה, נדרשת מעורבות הנהלת הארגון, לפעמים, גם בשלב ההתקנה. עקרונית, אין להשאיר את ניהוג פעילות הפיתוח והבקרה על המערכות בידי אנשי המקצוע בלבד. גם הנהלת הארגון חייבת להיות אחראית לפעילות זו.

ז. מערכות מידע ממוחשבות הנן מוצר עתיר עבודה. בשנות החמשים היו ההוצאות לציוד 75% מההשקעה בפיתוח מערכות לעומת שיעור של 25% לכוח־אדם, ואילו היום ההוצאות לציוד הן בשיעור של 25% לעומת 75% של הוצאות לכוח־אדם מכלל ההשקעה. מכאן החשיבות שיש ליחס למינוי האחראים לפיתוח ולבניית הצוות שיעסוק בפיתוח.

ח. תהליך הפיתוח של מערכות מידע ממוחשבות הנו תהליך איטרטיבי שתחילתו בתמונה הכללית והמשכו בירידה לפרטים. טבעי הדבר, שעם הירידה לפרטים מתברר, לעתים, שקיימת אי־התאמה בין הפרטים לבין המסגרת שנקבעה בשלב מוקדם ואז קיים הצורך לחזור לשלב הקודם וליישר את ההדורין. יש לשאוף שמספר האיטרציות (חזרות לשלב מוקדם) יהיה קטן ככל האפשר, באשר כל איטרציה הנה מקור לשינויים ועיכובים. יש לבדוק היטב את השפעת שינוי המיקרו על המקרו ולהיפך, וכמו־כן יש לוודא שהאיטרציות לא יתחילו בכל פעם מנקודת האפס. ברור שטיב העבודה בשלבים המוקדמים משפיע באופן מכריע על מספר האיטרציות.

ט. בעת הפיתוח של מערכות מידע ממוחשבות יש לבדוק מספר אלטרנטיבות ישימות ולא להסתפק בפתרון אחד. בדרך כלל בוחן הצוות המפתח מספר אלטרנטיבות ישימות, משווה ביניהן וממליץ על הטובה ביותר. במקרים קריטיים ניתן לתת אותה מערכת לבחינה לשני הצוותים. בחינתן של מספר אלטרנטיבות תתבצע רק בשלב ההגדרה.

י. יש לחלק את מערכת המידע שמפתחים לחוליות (מודולים). עקרון זה חל במיוחד על מערכות מידע מורכבות ובעלות היקף רחב. החלוקה לחוליות (או תת־מערכות) מאפשרת טיפול במרכיבים ברי־שליטה ומאפשרת פיתוח של צעד אחר צעד, דבר המקטין את הסיכון שבפיתוח.
עם זאת חייב האחראי על הפיתוח לראות כל הזמן את התמונה בשלמותה ולדאוג לכך, שבסופו של דבר יתחברו תת־המערכות למערכת אחת הרמונית שתשיג את מטרותיה ביעילות מירבית. יש להימנע מבנית חוליות בעלות היקף רחב.

יא. מאחר ומערכות מידע הנן מוצר מופשט, הדרך היחידה שבאמצעותה ניתן לחזות בתוצאות ולבקרן היא התיעוד (דוקומנטציה). לתיעוד התכונות הבאות:
— עצם הכתיבה והצורך בניסוח בהיר וחד־משמעי משמשים כלי לאיתור חולשות וחללים בתכנון מזה והוכחה לשלמות התכנון מזה.
— מאחר ובתהליך הפיתוח מעורב מספר רב יחסית של אנשים, יש צורך באמצעי תקשורת. התיעוד משמש כלי תקשורת מעולה.
— התיעוד הוא הכלי שבאמצעותו ניתן לנהל את עבודת הפיתוח, לבקר את הנעשה ולתכנן את שייעשה.
— התיעוד משמש גיבוי למערכת הנמצאת בתפעול ושתכניותיה עלולות להיהרס.
— התיעוד הנו כלי המאפשר תפעול ואחזקת המערכת. מפתחי המערכת אינם בהכרח אלה אשר יתחזקו ויפעילו אותה. ללא תיעוד לא יוכלו אלה שלא פיתחו את המערכת לתחזקה ולתפעלה.
— התיעוד הנו אמצעי הדרכה חיוני.
קיימים סוגים רבים של תיעוד. לכל סוג ייעודו שלו, מבנהו ושפתו של התיעוד. לכל שלב בתהליך הפיתוח תיעודו שלו, שהוא מוצרו של אותו שלב.

יב. התיעוד אינו כלי בקרה דפנסיבי או כלי תקשורת בלבד. תקני תיעוד משמשים כלי למיבנות (structuring) של תהליך הפיתוח. תקנים אלה משמשים כרשימת תיוג (check-list) לעוסקים בפיתוח ומאפשרים להם לבצע את עבודתם במסגרת מוגדרת, אחידה ומכוונת. תהליך הפיתוח המבוסס על תקני תיעוד מפורטים ושלמים הנו תהליך מוגדר מראש המונע את הצורך מלפתח לכל מערכת את מסגרת עבודתה.

יג. בתהליך הפיתוח משתתפים אנשים רבים בעלי רקע מקצועי ומעמדי שונה: הנהלת הארגון, המשתמשים, מרכז החישובים, מנתחי מערכות, תכניתנים, כותבי נהלים ואחרים. עבודה זו מכתיבה את הצורך במציאת מסגרת אר־גונית שתאפשר את עבודתם המשותפת.

יד. תכנון תהליך הפיתוח של מערכת מידע ממוחשבת הנו פעולה מורכבת. אולם, ללא תכנון מוקדם אין לגשת לביצוע. אם לא ניתן לתכנן את התהליך, סימן שאין אפשרות לבצעו. אין זה אומר, כמובן, שהתכנון אינו בר־שינוי לאורך זמן. ראוי לציין, כי תכנון עבודה שמומנט היצירה בה הוא בעל משקל, אינו פעולה קלה לביצוע. עצם החלוקה לשלבים של תהליך הפיתוח ותכנון לוח זמנים לכל שלב ושלב מאפשרים תכנון טוב יותר מאשר תכנון לגבי התהליך כשלמות.

טו. שני מוצרים הנם התולדה של תהליך התכנון של מערכת מידע ממוחשבת — תוכניות מחשב ונהלים למשתמשים. ללא שני מוצרים אלה אין התהליך מסתיים. חשיבותם של הנהלים אינה נופלת מחשיבותן של התוכניות. ללא נהלים כתובים בשפת הלקוח ומפורטים דיים — אין למשתמשים האפשרות להשתמש במערכת המידע. כדי לכתוב את הנהלים יש צורך במפרטים מטעמם של המפתחים וכן קיים הצורך בהדרכת המשתמשים לשימוש במערכת באמצעותם של הנהלים. את הנהלים יש לנסות כפי שמנסים את התוכניות.

טז. פעולת ההסבה מהמערכת הקיימת למערכת החדשה היא מערכת בפני עצמה, שלגביה יש צורך בתכנון ובקרה, כתיבת תוכניות ונהלים, הדרכה וכו', כלומר, שקיימות למעשה שלוש תת־מערכות: המערכת הקיימת, מערכת ההסבה והמערכת החדשה. ההחלטה לעבור משלב הפיתוח לשלב ההסבה הנה החלטה קריטית. אין לעבור לשלב ההסבה לפני שיתברר לחלוטין שתהליך הפיתוח אמנם הסתיים במלואו.

יז. כל ארגון העוסק בפיתוחן של מערכות מידע ממוחשבות חייב לפתח לעצמו תוכנית־אב. מטרותיה של תוכנית האב הן:

— להקיף את תוכנית הפיתוח לטווח הבינוני והארוך ובאמצעותה להכין את המשאבים הדרושים לביצועה.
— לקבוע את סדר הפיתוח האופטימלי.
— לפתח את המערכות כך, שבסופו של דבר, במקום שהדבר נדרש, ניתן יהיה לשלב מערכות.
— למנוע פיתוח "פראי" של מערכות מידע ללא קשר לתוכנית כוללת, הלוקחת בחשבון את צרכי הארגון ומטרותיו.

הקריטריונים העיקריים שיש לקחת בחשבון בעת קביעת סדר העדיפות של פיתוח המערכת הם:

— הסדר הלוגי של שלבי הפיתוח.
— יכולת הארגון לקלוט את המערכת.

— זמנו ומחירו של הפיתוח.
— האם המערכת פותרת בעיות בשטחים החשובים לארגון.
— התמורה הפוטנציאלית (הישירה והעקיפה) לארגון מפיתוחה של המערכת.
— אילו מערכות פועלות ומפותחות אצל ארגונים מתחרים.
— יכולת הפיתוח של הארגון (כוח־אדם מקצועי, טיבו וכמותו).
— האפקט הפסיכולוגי (הצורך ברכישת אמונו של הארגון על־ידי היחידה המפתחת את המערכת).

ניתן להבדיל בקבוצת הקריטריונים המפורטת לעיל בין קריטריונים כמותיים לבין קריטריונים איכותיים. חייבת, כמובן, להיות שיטה לשימוש מושכל בקריטריונים.

יח. תהליך הפיתוח של מערכות מידע ממוחשבות צריך לקחת בחשבון את צרכיו ויכולתו של המשתמש ובה־בשעה גם את צרכיה ויכולתה של היחידה שתפעיל ותאחזק את המערכת. היחידה המתפעלת הנה בבחינת בית־חרושת שפעולותיו צריכות להיות מוגדרות, רוטיניות וחד־משמעיות ככל האפשר. נוסף על כך, כל מערכת מידע ממוחשבת עוברת תהליך של שינויים ותוספות והיא צריכה להיות בנויה כך, שתהליך זה יוכל להתבצע בצורה המיטבית.

יט. לא כל מרכיביה של מערכת מידע הם ממוחשבים. כל מערכת כזו מור־כבת ממרכיבים ממוחשבים ומרכיבים ידניים, הקשורים ביניהם ומהווים יחדיו מערכת אחת הרמונית. משמעות עובדה זו היא, שהאחראי לפיתוח מערכות המידע חייב בו־זמנית לשלוט על פיתוחם של מרכיבים ממוחשבים וידניים.

כ. חוק התפוקה השולית הפוחתת תקף בכל הקשור לפיתוחן של מערכות מידע. תוספת עובדים לצוות האחראי לפיתוח יכולה לתרום משמעותית לקצב הפיתוח רק עד גבול מסוים. הגדלתו של הצוות אל מעבר לגבול זה אינה מיטבית והתפוקה השולית שלו תפחת ככל שיגדל הצוות. במלים אחרות, הגדלתו של הצוות העוסק בפיתוח אינה תורמת בכל מקרה להחשת קצב הפיתוח.

תהליך הפיתוח וההפעלה — מרכיביו ומבצעיו

תהליך הפיתוח וההפעלה של מערכות מידע ממוחשבות מתחלק כאמור לארבעה שלבים עיקריים, ואלה מצדם מתחלקים לשלבי־משנה כפי שיפורט להלן. פיתוח המערכות חייב להיעשות בסדר השלבים בצורה שיטתית וללא פסיחה על שלבים. המשקל הסגולי של השלבים השונים אפשר שישתנה מארגון לארגון

וממערכת למערכת, אולם אין לשנות את הסדר ואת השלמות הצריכים לאפיין את פיתוחן של כל המערכות.

לכל שלב הפונקציה המבצעת אותו (מנתח מידע, מעצב מערכת, תוכניתן וכו׳). ייתכנו מקרים שבהם מספר פונקציות מבוצעות על־ידי אותו אדם. מנתח המידע ומעצב המערכת יכולים למשל, להיות אותו אדם.

אחת מבעיות היסוד של תהליך הפיתוח הנה המימשק (interface) בין השלבים, במיוחד כאשר האחריות לביצוע עוברת לאדם אחר (לדוגמא, במעבר משלב תכנון המערכות לשלב התכנות). האחראי על הפיתוח חייב לוודא מקרוב, שאכן המעבר נעשה באופן חלק ושלם ושכל המידע הועבר מהאחראי האחד לשני, הן בכתב (תיעוד) והן בעל־פה. קיימת, אפוא, תקופת מעבר בין שלב אחד למשנהו, שבה עובדים האחראים על השלבים בצוותא כדי לשמור על רציפותו של תהליך הפיתוח.

כל שלב יוצר פלט המשמש כקלט לשלב הבא אחריו. הפלט לפני הפיכתו לקלט חייב לעבור ביקורת שתקיף את הנקודות הבאות:

— הצד התיפקודי של המערכת.
— הצד הכלכלי, היישומי והטכני.
— לוח זמנים עתידי, המשאבים הדרושים ויכולת הקצאת המשאבים לסיום הפיתוח.

לסיכום, חלוקת תהליך הפיתוח למספר שלבים ותת־שלבים מאפשרת בקרה על התהליך על־פי קטעים ברי־שליטה (מבחינת ההשקעה, המוצר ולוח הזמנים), האצלת האחריות לבעלי המקצוע המתאים, ביצוע איטרציות דרושות בצורה מיטבית ובדיקת הכדאיות הכלכלית של המערכת באופן פרוגרסיבי ובר־שליטה.

מחזור החיים של מערכת ממוחשבת בנוי מהשלבים הבאים:

הגדרה

1. ניתוח ראשוני — הגדרת הדרישות למידע וקביעת היקף מערכת המידע וגבולותיה.
2. חקר ישימות — קביעת ישימותה של המערכת המוצעת מהב־חינות הבאות: כלכלית, טכנית ותפעולית.
3. ניתוח המידע — בניית המפרטים התפקודיים המפורטים של המערכת החדשה וכן קביעת דרך המעבר (הסבה) מהמערכת הקיימת לחדשה.
4. תכנון מערכת המידע הממוחשבת — תכנון המפרטים לתכנות, להסבה ולכתיבת הנהלים.

בנייה

5. תכנות — קידוד התוכניות וניסוין.
6. כתיבת הנהלים.

התקנה

7. הסבה — הבאת המערכת החדשה לפעולה תקינה.

תפעול

8. תפעול ואחזקה.
9. בקרה — בקרה שוטפת ושיטתית על המערכת.
10. סיום — החלטה לחסל את המערכת או להחליפה בחדשה.

חלוקה זו הנה אחת מני רבות. עם זאת, כל החלוקות שוות בעקרון וההבדלים הנם, בעיקר, הבדלים ברמת הפירוט.

להלן נתונים טיפוסיים על פילוג ההשקעה וזמן הפיתוח באחוזים בין שלבי תהליך הפיתוח של מערכת מידע ממוחשבת:

השלב	ההשקעה	זמן הפיתוח
ניתוח ראשוני	2%	5%
חקר ישימות	1%	5%
ניתוח המידע	10%	15%
תכנון מערכת המידע הממוחשבת	8%	10%
תכנות וכתיבת נהלים	40%	35%
הסבה — התקנה	39%	30%
	100%	100%

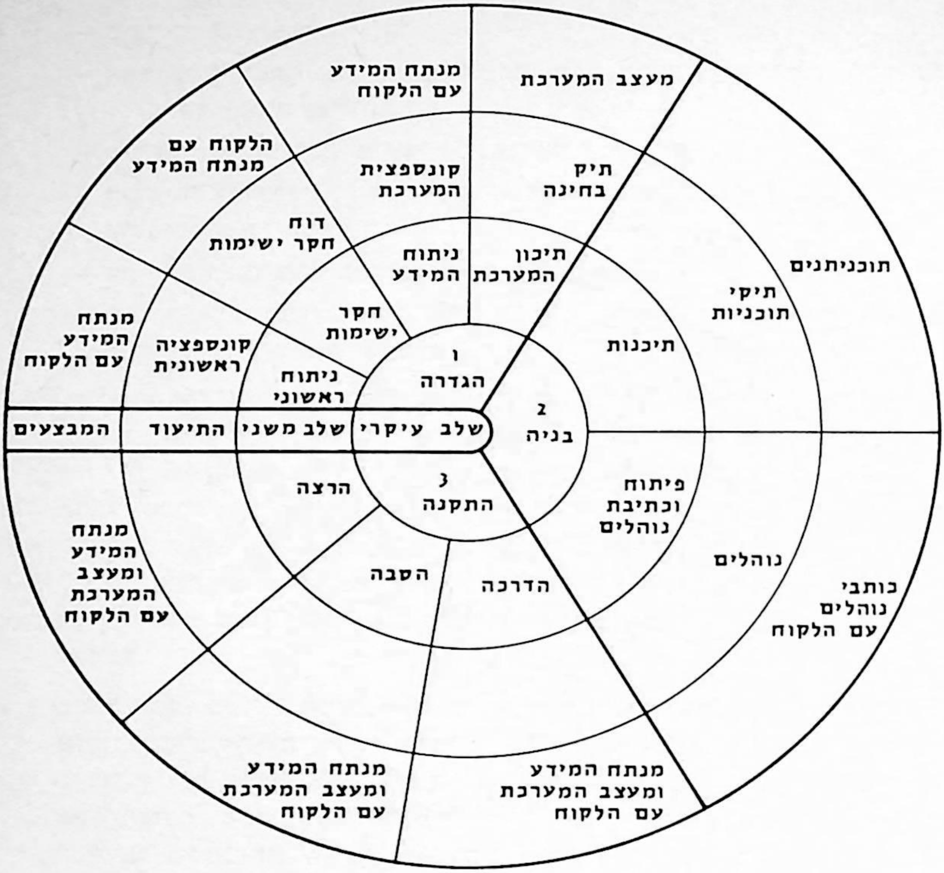
טבלה 3.1 מתארת את השלבים ומטרותיהם, מי מבצע כל שלב, מי מאשר את המעבר משלב אחד למשנהו וכן את התיעוד שהוא המוצר הסופי של כל שלב. תיאור גראפי של הטבלה מוצג בתרשים 3.10.

טבלה 3.1 מחזור החיים של מערכת מידע ממוחשבת

השלב	שם השלב	מטרה ותוכן	המבצעים	אישר מעבר לשלב הבא	תיעוד
1	ניתוח מידע ראשוני (קביעת הדרישות למידע)	1. קביעת הדרישות למידע כפועל יוצא מפעילויות הארגון. 2. קביעת מקורות הקלט. 3. קביעת היקף המערכת וגבולותיה. 4. קביעת מקומה של המערכת בארגון ואופן פעולתה בו.	מנתח המידע עם הלקוח	הלקוח	קונספציה ראשונית (בשפת הלקוח)
2	חקר ישימות (Feasibility Study)	— קביעת ישימותה של המערכת המוצעת מהבחינות הבאות: — כלכלית: עלות מול תועלת — טכנית — תיפעולית — קביעת כדאיות המיחשוב.	מנתח המידע עם הלקוח	הלקוח	דוח חקר־ישימות (בשפת הלקוח)
3	ניתוח המידע (System Specificationes)	1. בניית המפרטים התיפקודיים המפורטים של המערכת החדשה וכן דרך המעבר מהמערכת הקיימת לחדשה. 2. בניית מפרטים טכניים — פירוק המערכת למרכיביה וקביעת נתוני המרכיבים. 3. בדיקה חוזרת של הניתוח הכלכלי שנעשה בשלב 2.	הלקוח עם מנתח המידע	הלקוח	קונספצית המערכת (בשפת הלקוח)

השלב	שם השלב	מטרה ותוכן	המבצעים	אישור מעבר לשלב הבא	תיעוד
4	תיכון מערכת המידע הממוחשבת (System Engineering)	1. תיכנון המפרטים לתיכנות ולהסבה. 2. בדיקה חוזרת של הניתוח הכלכלי שנעשה בשלב 2. 3. תיכנון מפרטים לנהלים.	מעצב המערכת	האחראי על פיתוח המערכות והלקוח	מפרטים לתכנות (תיקי בחינה)
5	תיכנות (Programming)	1. תיכנות המערכת. 2. ניסוי התוכניות (כל אחת בפני עצמה). 3. תיעוד התוכניות. 4. ניסוי מערכת התוכניות (כשלמות אחת).	תכניתנים	תכניתן ראשי ומעצב המערכת	1. תיקי תוכניות 2. תיקי הפעלה לחדר המחשב 3. תיקי הוראות לניקוב.
6	פיתוח וכתיבת נהלים (procedures design)	פיתוח נהלים שיאפשרו פעולה מסודרת ויעילה במעבר למערכת החדשה ובתפעולה השוטף.	הלקוח עם מנתח המידע ומעצב המערכת	הלקוח	1. נהלים למשתמש. 2. נהלים לממלאי הקלט. 3. נהלים לעובדי קלט/פלט 4. נהלי הסבה.
7	הסבה (conversion)	הבאת המערכת החדשה לפעולה תקינה: א. הדרכת כל הנוגעים בדבר ב. בניית הקבצים החדשים ג. ריצה במקביל (אם יש צורך) ד. עריכת שינויי התאמה	הלקוח מנתח המידע מעצב המערכת תכניתנים מפעילים עובדי קלט/פלט	הלקוח	

השלב	שם השלב	מטרה ותוכן	המבצעים	אישור מעבר לשלב הבא	תיעוד
8	תפעול ואחזקה (operation and maintenance)	1. הפעלת המערכת בהתאם למתוכנן. 2. בצוע שינויים ותוספות במטרה לשפר את המערכת ולהיענות לדרישות נוספות.	מפעילים אנשי קלט/פלט לקוח מנתח המידע מעצב המערכת תכניתנים		
9	בקרה (post audit)	בקרה שוטפת באמצעות מדדי ההצלחה של המערכת (עלות מול תועלת)	לקוח ומנתח המידע	הלקוח	דוחות בקרה שוטפים
10	סיום	החלטה שמערכת המידע סיימה את תפקידה: א. לבנות חדשה ב. לחסל את המערכת.	לקוח ומנתח המידע	הלקוח	



תרשים 3.10: הצגה גראפית של תהליך הפיתוח של מערכת מידע ממוחשבת

פירוט שלבי הפיתוח

1. התוצאה של שלב ניתוח המידע הראשוני תכיל את הנתונים הבאים:
   — פירוט הבעיה והגדרתה.
   — פירוט צרכי המידע שהם פועל יוצא של הבעיה.
   — נמעני המידע.
   — תדירות קבלת המידע.
   — פירוט התנועות שיעובדו (הקלט למערכת).
   — היקפה וגבולותיה של המערכת שתפיק את המידע.

2. שלב חקר הישימות חייב לספק תשובות לשאלות הבאות:
   — מהן הבעיות שהמערכת אמורה לפתור (האם הבעיות הן בעיות והאם דרגת חשיבותן מצדיקה פיתוחה של המערכת המוצעת).

— כיצד ייפתרו הבעיות.
— האם ניתן לפתח את המערכת מהבחינה הטכנית ומבחינת המשאבים לפיתוח (כמות ואיכות).
— איך תתקבל המערכת על־ידי המשתמשים המיועדים.
— הכדאיות הכלכלית של המערכת — האם היחס בין עלות לתועלת מצדיק את פיתוחה של המערכת.
— מה יהיה הצעד הבא בתהליך פיתוח המערכת.
— האם תואמת המערכת את תוכנית האב ואת סדרי העדיפויות שקבעה ועדת ההיגוי (או הנהלת הארגון).

3. בשלב ניתוח המידע ייבנו המפרטים התיפקודיים המפורטים של מערכת המידע. המערכת שהוגדרה בשלב הניתוח הראשוני תפורק למרכיביה (חוליות או תת־מערכות) כשהקשרים ביניהם יוגדרו במפורט. כמו כן יתבצע תכנון טכני של המערכת ברמת פירוט המאפשרת בדיקת יכולתה של המערכת להיות מיושמת מהבחינה הטכנית. ולבסוף, תיעשה בדיקה חוזרת של הכדאיות הכלכלית של המערכת. משום היקפו וחשיבותו מתחלק שלב זה לשלוש: הבנת המערכת הקיימת, קביעת הדרישות ותכנון המערכת החדשה.

4. בשלב תכנון מערכת המידע הממוחשבת יתבצע תרגום המפרטים התיפקודיים שנבנו בשלב הקודם בשפת הלקוח, עבורו ועם המבט אל הארגון, לשפת התכניתן ועם המבט אל המחשב. המרכיבים שאותם הוחלט למחשב יקבלו את עיצובם הממוחשב האופטימלי ויקבעו המפרטים לתכנות ולכתיבת הנהלים. בשלב זה תפורק המערכת לתוכניות. מפרטים אלו יכילו את הנושאים הבאים לגבי כל תוכנית:

— קלט: מבנה מסמכי המקור, מקורם, קצבם ונפחם וכן צורת המעבר למסמכים הנקראים על־ידי המחשב.
— קבצים: אלו קבצים, אמצעי האחסנה, שיטת העיבוד, מבנה הקבצים (הרשומות) ונפחם (לאורך זמן).
— עיבוד: כללי העיבוד, קצב העיבוד.
— פלט: אלו דוחות, מה צורתם ומה צורת הפקתם וקצבם.
— ניסוי והתקנה: נתוני הניסוי, צורת הניסוי ומדדים לבדיקת טיב הניסוי של התוכניות, צורת בניית הקבצים החדשים וצורת בדיקת תוכנם.

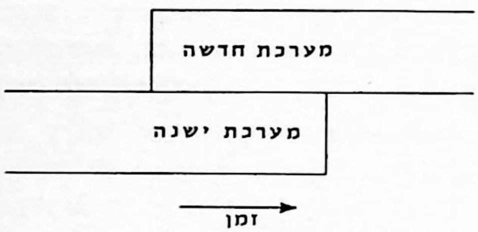
תיק הבחינה שהוא התוצר של שלב זה יהיה בנוי בצורה הבאה:
— תרשימי זרימה של המערכת ושל התוכניות שיתארו את תזרים העיבוד, הקלטים, הפלטים, הקבצים והציוד שבו ישתמשו. דגש יושם על תיאור הקשרים בין התוכניות.

— מטריצות שיראו את הקשר בין נתוני הקלט, הקבצים והפלט.
— מערך טופסי המקור, טופסי הקלט למחשב, קבצים ופלט.
— תיאור כל בדיקות התקינות, הסבירות והשלמות.
— תיאור מילולי של כל תוכנית, תיפקודה וכללי העיבוד המתבצע בה.
— פירוט המפרטים לכתיבת נהלי ההפעלה והשימוש.
— תוכנית לבדיקתה של כל תוכנית ושל המערכת כולה.
— תוכנית לבניית הקבצים, או על־ידי איסוף נתונים אצל המשתמש, או על־ידי הסבתם מקבצים קיימים.
— תכנון התקנתה של המערכת.
— תכנון עבודת התכנות וכתיבת הנהלים (לוח זמנים והקצאת משאבים).

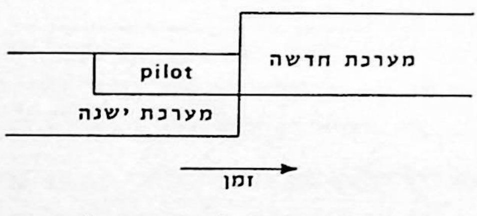
5. בשלב ההסבה יתבצעו ההדרכה וההכשרה של המתפעלים את המערכת ושל המשתמשים בה, הרצת המערכת והתקנתה בארגון (שתילת המערכת בארגון).

קיימות 4 שיטות עיקריות לביצוע המעבר מהמערכת הישנה למערכת החדשה:

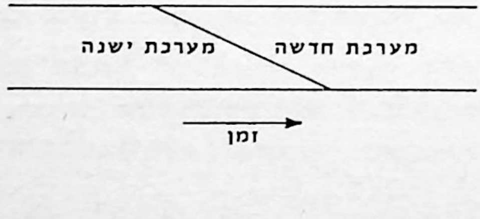
א. ריצה במקביל:



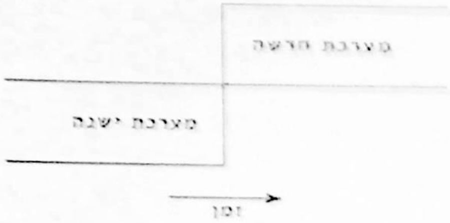
ב. מעבר דרך מערכת נסיונית (pilot):



ג. מעבר הדרגתי (phase-in) — חוליה אחר חוליה:

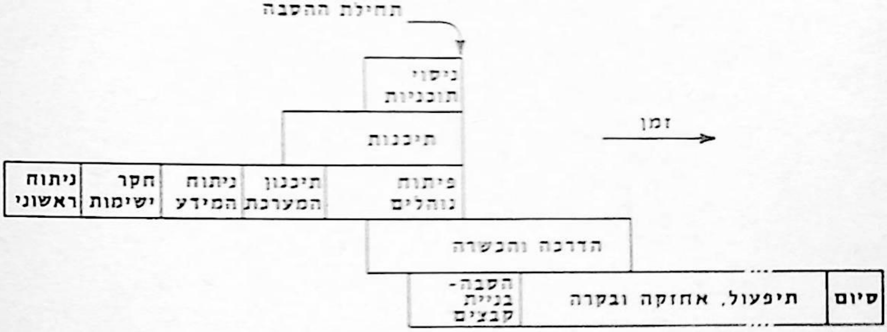


ד. מעבר חד מהמערכת הישנה לחדשה:



בכל מקרה יש לשקול איזו שיטת הסבה היא הטובה ביותר; ייתכן שילוב בין השיטות.

6. סיום חייה של המערכת (ובדרך כלל גם תחילת חייה של מערכת חדשה — מכאן המושג מחזור החיים) בא לידי ביטוי באותו שלב של חיי המערכת שבו הדרישות לשינויים מחד ומספר השגיאות (bugs) במערכת, מאידך, מגיע לרמה כזו, שלא כדאי להמשיך ולהפעיל עוד את המערכת הישנה.
שלביו של תהליך הפיתוח של מערכות מידע ממוחשבות אינם מתבצעים דווקא בטור. מספר שלבים יכולים להתבצע במקביל. מאידך ישנם שלבים שבהם לא ניתן להתחיל בטרם הסתיים השלב הקודם להם. ביצוע השלבים על פני זמן מתואר בתרשים 3.11.



תרשים 3.11: שלבי הפיתוח וההפעלה על פני הזמן

יש להזכיר שוב, שהתיעוד הוא המוצר של תהליך הפיתוח. הוא נוצר בעת תהליך הפיתוח וסיום כתיבתו מסמל את סיום הפיתוח (כל שלב והתיעוד שלו). מבחינים בארבעה סוגי תיעוד עיקריים:

א. *תיעוד המערכת המתאר את המערכת כולה* — נכתב על־ידי מנתחי מערכות (מנתח המידע ומעצב המערכת).

תיעוד המערכת מכיל כמה סוגים משניים של תיעוד:

תוכנית האב — תיעוד כללי של כל מערכות המידע המתוכננות והפועלות.
קונספציה — תיאור מערכת המידע בשפתו של המשתמש ועבורו.

תיק בחינה — תיאור מערכת המידע על מרכיביה (התוכניות) בשפתו של התכניתן ועבורו.

ב. תיעוד התוכניות המכיל את פרטי כל תוכנית ותוכנית במערכת — נכתב על־ידי התכניתנים.

ג. תיעוד למתפעלי המערכת — מפעילי המחשב, אנשי קלט/פלט, נקבניות וכו׳ — נכתב על־ידי התכניתנים ומנתחי המערכות.

ד. הנהלים למשתמשים במערכת — נכתב בידי מנתחי המערכות בשיתופם הפעיל של המשתמשים.

ארגון עבודת הפיתוח של מערכת מידע ממוחשבת

מערכת מידע ממוחשבת הנה כלי בידי הארגון שעבורו היא נבנית. בארגון היא משמשת כלי הן בידי המערכת התפעולית והן בידי המערכת הניהולית. מערכת המידע חייבת להשתלב בארגון, להישתל בו, להיפך לחלק ממנו ולייצג את צרכיו של הארגון. כמו־כן, מדובר ב״מוצר״ שלשם יצירתו הושקעו משאבים רבים. המסקנה בעקבות האמור לעיל היא, שקיים צורך במעורבותה הפעילה של הנהלת הארגון בתהליך הפיתוח של מערכות המידע. אולם, הנהלת הארגון יכולה להיות מעורבת בעיקר בתמונה הכללית של תהליך הפיתוח ונדרשת, אפוא, גם מעורבות של המשתמשים שעבורם נבנית המערכת. מעורבות זו אינה יכולה להצטמצם לתמונה הכללית בלבד. כדי שזו תבוא לידי ביטוי בצורה יעילה ושיטתית, יש צורך בקביעת מסגרות ארגוניות מתאימות, שיאפשרו את המעורבות הדרושה.

ברור שצורת המעורבות תהיה שונה בכל שלב משלבי הפיתוח. צורת הארגון המוצגת להלן הנה אחת מני רבות, אולם הוכיחה עצמה בארגונים רבים. בצורה זו תפקידיו של הגוף המנהל את פיתוח מערכות המידע בארגון הם:

— הגדרת גבולותיה של המערכת (הפרויקט).
— קביעה וכימות של מטרות ביצועי המערכת והחלטה על כדאיות הפיתוח של המערכת.
— תכנון כוללני של תהליך הפיתוח והקצאת משאבים לתהליך.
— בקרה על תהליך הפיתוח הן מבחינת השקעת המשאבים בפועל מול התקציבים המאושרים והן מבחינת לוח הזמנים.
— הערכת ביצועי המערכת החדשה מול מטרותיה.

הגוף המנהל את פיתוח מערכות המידע מורכב משלושה תת־גופים:

א. ועדת היגוי שבה מיוצגים המנהל הכללי וסגניו אשר להם קשר למערכות המידע. לוועדה זו, שהיא המוסד העליון בגוף המנהל, האחריות הכוללנית לפיתוח המערכות.

ב. מנהל היחידה המקצועית העוסקת בפיתוח מערכות המידע האחראי לתהליך הפיתוח מבחינתו התפעולית היומיומית ומבחינתו המקצועית.

ג. קבוצת הפרויקט (project team) שבה נציגים (מתאמים) מטעמם של המשתמשים וכן נציגים מטעמם של המפתחים. קבוצה זו מאפשרת ביצוע תהליך הפיתות כמאמץ משותף של היחידה המקצועית והמשתמשים. רצוי כמובן, שנציגי המשתמשים בקבוצת הפרויקט יקדישו את כל זמנם — במיוחד בשלב ההגדרה — לעבודה זו. נציגים אלה אמורים לייצג את המשתמש, את בעיותיו, דרישותיו וצרכיו. עם תום הפיתוח, יהיו הם אלה שיתקינו את המערכת בארגונם. רצוי שראש קבוצת הפרויקט יהיה נציג המשתמש ולא בעל מקצוע מהיחידה המפקחת. קבוצת הפרויקט הנה הצוות שהוקצה למאמץ הפיתוח של המערכת ואחראית לאיכותה וליעילותה של המערכת.

הצורך בוועדת היגוי שבה מיוצגת הנהלת הארגון חשיבותו בכך שגבולותיהן של מערכות המידע אינם תואמים בהכרח את המבנה הארגוני הפורמלי. ניתן לצרף לוועדת ההיגוי יועץ חיצוני בעל ניסיון וידע שיוכל לשמש בתפקיד מבקר פנימי. לשם יעילות העבודה רצוי שוועדת ההיגוי לא תהיה רבת חברים.

תפקידיה המפורטים של וועדת ההיגוי הם:

- מתן הנחיות לצורת ביצוע חקר הישימות.
- השתתפות פעילה בשלב חקר הישימות.
- החלטה על התחלה, המשך, או הפסקה בפיתוח.
- הגדרת מטרות המערכת, הן איכותית והן כמותית.
- אישור הזמנת ציוד מחשבים.
- אישור המשאבים שיוקצו לפיתוח.
- גיבוי לפעולת הפיתוח כלפי הארגון.
- אישור נהלי תכנון ובקרה של תהליך הפיתוח.
- בקרה על השימוש במשאבים ועל קצב הפיתוח.
- קביעת סדר הפיתוח של מערכות המידע.
- ביצוע בקרה על המערכות הפועלות.

רצוי שהיוזמה לפיתוחן של מערכות המידע תבוא מצד המשתמש דרך וועדת ההיגוי. קיומה של וועדת היגוי מבטיח, שנקודת המוצא בכל הקשור למערכות המידע תהיה הארגון וצרכיו ולא הצד הטכני־המחשבי.

.3
גישות לבניית מערכות מידע

מבוא

קיימות מספר גישות עקרוניות לבניית מערכות מידע. יש להבחין בין הגישות לבניית תוכנית האב של מערכות המידע עבור ארגון (או תת-ארגון) לבין הגישות לבניית מערכת מידע מסויימת. אבחנה זו אינה עקרונית, כפי שיובהר להלן.

תוכנית אב לבניית מערכות מידע הנה תוכנית כוללנית, המקיפה את צרכי המידע השונים של הארגון, מתחמת אותם למערכות וקובעת את יחסי הגומלין ביניהם.

תוכנית האב משמשת כבסיס לתכנון עבודת פיתוח מערכות המידע בארגון לטווח הבינוני והארוך. מאחר ואנו מצויים בסביבה של משאבים מוגבלים, מתבצעת עבודת הפיתוח בשלבים, לפיהם סדר הפיתוח הנו פועל יוצא מצרכיו של הארגון, יכולתו לפתח ויכולתו לקלוט ולתפעל את המערכות.

פיתוחה של מערכת מידע מסוימת, כשהיא מהווה חלק מתוכנית האב, מתבצע במסגרת כללית נתונה, לאמור: גבולותיה של המערכת ויחסי הגומלין שלה עם שאר המערכות נתונים למעשה על-ידי תוכנית האב. עם זאת, המבנה הפנימי של המערכת אינו נתון.

גישות לבניית תוכנית אב למערכות מידע בארגון

א. גישה בלתי מתוכננת

פיתוח המערכות בארגון מתבצע ללא תכנון מוקדם אלא בהתאם לצרכים המתעוררים חדשות לבקרים. זוהי גישת אד-הוק וכיבוי-שרפות, שאינה עושה שימוש בתוכנית אב. תוצאותיה של גישה זו הן "אי-מיכון" ומערכות

מידע שאינן מיטביות. גישה זו יכולה להיות מוצדקת רק במקרים שבהם הארגון נמצא בתהליך שינוי מתמיד. עם התייצבותו של הארגון, עוברות המערכות שנבנו לאור גישה זו, בדרך כלל, תיכון מחדש.

ב. גישת בסיס הנתונים

מהותה של גישה זו הנה איסופם, רישומם, אגירתם ועדכונם של הנתונים הקיימים בארגון מבלי לבצע ניתוח מוקדם בכל הקשור לשימוש שניתן לעשות בנתונים אלה. בהתעורר הצורך במידע מסוים, מתבצע עיבוד הנתונים הדרוש לשם הפקת מידע זה.

למערכות הבנויות בהתאם לגישה זו קוראים: input oriented systems.

הצידוק לשימוש בגישה זו הנו הטענה שלא ניתן לחזות את צרכי המידע של הארגון, והדרך היחידה האפשרית היא, אפוא, איסוף הנתונים ואגירתם וכן אספקת כלים לעיבודם ולשליפתם בעת הצורך.

גישה זו אינה מעשית משום שלא ניתן לאסוף, לאגור ולעדכן את כל נתוני הארגון ומשום שהמשתמש חייב בכל מקרה שבו הוא זקוק למידע לדרוש עיבוד מיוחד כדי להפיק אותו.

ג. הגישה האבולוציונית או מלמטה למעלה (bottom up)

מהותה של גישה זו הנה בניית תוכנית האב מכיוון האירועים והפעולות הבסיסיים המתבצעים בארגון. כלומר, שלב הבנייה הראשון הוא בניית מערכות מידע שיפיקו מידע לתיפעול בלבד ורק בשלבים מאוחרים יתפתחו המערכות כך שיוכלו להפיק גם מידע לניהול. שלבי הביצוע לפיתוח מערכות המידע בארגון בשיטה זו הם:

1. מערכות מידע מבודדות בעלות קבצים יחודיים. המידע המופק הוא לתיפעול.
2. שילוב הקבצים היחודיים, במקום שנדרש, לבסיס נתונים.
3. הוספת מודלים לתכנון ולעשיית החלטות שיאפשרו הפקת מידע לניהול לדרגי הניהול הנמוכים והבינוניים.
4. שילוב המודלים לתכנון ולעשיית החלטות למסגרת של בסיס מודלים (model base), כך שיוכלו לעמוד לרשותן של כל מערכות המידע וכן הרחבת בסיסי הנתונים עם נתונים חיצוניים. במצב זה יכולות מערכות המידע לתמוך גם בפעילויות הניהול של דרגי הניהול הגבוהים.

גישה זו הנה גישה אבולוציונית המאפשרת התאמת המערכות לצרכים הממשיים של הארגון בשיטת הצעד אחר צעד. עם זאת, מגבלתה העיקרית נעוצה בחוסר יכולתה לראות מראש את הצרכים הכוללים של הארגון, לפיכך שילוב המערכות אינו מיטבי. יתר על כן, צרכים שאינם מועלים במפגיע על־ידי הצרכנים אינם נענים. אימוצה של גישה זו מביא בהכרח לצורך בתיכון מחדש של מערכות המידע בעקבות הגידול שחל בהן. זאת,

משום שהתיכון הראשוני אינו יכול לקחת בחשבון את ההיקף העתידי של המערכות.

ד. גישת המלמעלה למטה (top down)

מהותה של גישה זו הנו איתור צרכי המידע הכוללניים של הארגון ותיכון מערכות המידע בהתאם לצרכים כוללים אלה. איתור הצרכים הכוללניים לפני תיכון מערכות המידע מאפשר תיחום ושילוב המערכות.

תהליך איתור צרכי המידע הכוללניים מתבצע בדרך הבאה:

1. הגדרת מטרות הארגון.
2. ניתוח סביבת הארגון ואילוציה.
3. הגדרה וניתוח של הפעילויות והתהליכים המתבצעים בארגון, לאור מטרותיו.
4. איתור החלטות המפתח, לסוגיהן השונים, אשר יש לקבלן במסגרת הפעילויות והתהליכים.
5. איתור המידע הדרוש לתהליכי קבלת ההחלטות ולביצוע הפעילויות.
6. בניית זרימת המידע בארגון.
7. תיחום המידע למערכות והגדרת קשרי הגומלין ביניהן.

המעלה העיקרית של גישה זו היא השיטתיות והראייה הכוללת המאפשרות בניית תוכנית אב מיטבית.

המגבלה העיקרית שלה היא היקפה ומורכבותה והקושי בביצועה.

ה. הגישה המשולבת

הגישה המשולבת מנסה להתגבר על מגבלותיהן של גישת המלמעלה למטה והמלמטה למעלה, ולשמור, במידת האפשר, על מעלותיהן.
בגישה זו נבנית רק המסגרת של תוכנית האב לפי גישת המלמעלה למטה ואילו סדר יצירת הבנייה של המערכות הבודדות, בתוך מסגרת זו, מתבצע בהתאם לגישת המלמטה למעלה.

שתי גישות לבניית מערכות מידע מסוימות

המושג גישות לבניית מערכות מידע מסוימות מגדיר את הדרך שבה נוקטים להגדרת צרכי המידע שאותם צריכה המערכת לספק. קיימות שתי גישות עיקריות:

א. ניתוח החלטות.
ב. ניתוח נתונים.

אלו דומות בעיקרן לגישת המלמעלה למטה ולגישת המלמטה למעלה. למעשה, אין גישות אלו עומדות כל אחת בפני עצמה. שילובן הנו שיטת בנייה מעשית ההולמת את מרבית המקרים.

א. גישת ניתוח החלטות

גישה זו מנסה לאתר את צרכי המידע אשר צריכה לספק מערכת מידע מסוימת על־ידי ניתוח תהליכי קבלת ההחלטות שבהן אמורה המערכת לתמוך. דרך הניתוח היא כדלהלן:

1. זיהוי ואיתור ההחלטות.
2. הגדרה וניתוח של תהליכי קבלת ההחלטות (כולל המודל או האלגוריתם שבו עושים שימוש).
3. זיהוי המידע הדרוש לתהליך.
4. קביעת הנתונים וצורת עיבודם.

ב. גישת ניתוח נתונים

גישה זו מנסה לקבוע את צרכי המידע שאותם אמורה מערכת מידע מסוימת לספק על־ידי ניתוח הנתונים והמידע הקיימים ושיהיו קיימים בארגון. דרך הניתוח היא כדלהלן:

1. איסוף מסמכים, דוחות, קבצים וכו׳ שבהם עושים שימוש בארגון וזיהוי הנתונים והמידע הכלולים בהם.
2. ראיון צרכני המידע וניתוח מערכות מידע דומות שבהן עושים שימוש בארגונים אחרים, במגמה לאתר נתונים ומידע נוספים פוטנציאליים, שבהן עשוי הארגון לעשות שימוש.
3. תיכון מערכת מידע שתספק את הנתונים והמידע שזוהו והוגדרו.

גישה זו משמעותה התבססות על הצרכים כפי שהם באים לידי ביטוי בהווה או על־ידי בעלי התפקידים בארגון ולא כפועל יוצא ממטרותיו ופעילויותיו של הארגון.

.4

מערכות מידע משולבות

מערכות מידע משולבות (integrated information systems) הן בראש ובראשונה מערכות מידע. מערכת מידע, כמו כל מערכת, ניתנת לתיכון בדרגות שונות של עצמאות החלקים השונים של המערכת. מבחינה היסטורית, נדרשו תת־מערכות מידע, שלא כמערכות אחרות, לעצמאות ניכרת. קיבולת עיבוד הנתונים שעמדה לרשות מתכנני מערכת המידע היתה מוגבלת בגלל הטכנולוגיה הקיימת, כה מוגבלת, שניתן היה (לפחות היסטורית) להקים מערכות כאלה רק על בסיס חלקי. במערכת מידע חלקית כזאת כל תת־מערכת היתה אחראית לאיסוף הנתונים שלה. תיאום איסוף הנתונים בין תת־המערכות בתוך הארגון דרש אמצעים יותר מתקדמים של טכנולוגית עיבוד נתונים ואמצעים יותר מתקדמים של מתן גישה לנתונים שנמצאו בקבצים שונים. לפיכך, היה זה זול יותר (שוב מבחינה היסטורית) להכפיל את איסוף הנתונים, כלומר, לבצע עבודה כפולה של איסוף נתונים עבור יישומים או תת־מערכות שונות, במקום לבצע את התיאום שנדרש למניעת אותה הכפלה או יתרות של איסוף נתונים. רוב המערכות הממוחשבות תוכננו בגישה של מערכות העיבוד הידניות, שבהן כל יישום היה מעובד בנפרד על־ידי שימוש בקבצים נפרדים. גישה זו היתה אמנם יעילה מבחינת העיבוד ומבחינת הבקרה, אולם הובילה להכפלה של קבצים, כיוון שלחלקים של אותם קבצים יש שדות נתונים שחזרו על עצמם בקבצים שונים, והנתונים אף הם אינם בהכרח זהים. פירוש הדבר כי בגישה הישנה, שהיתה מבוססת על העיבוד הידני, כאשר עברו ליישום שהשתמש בנתונים מכמה יישומים שונים, הכרח היה להקים קובץ מיוחד לאותו יישום הכולל בתוכו נתונים של קבצים אחרים.

כאמור, נבע הדבר, בין היתר, גם מדרגת ההתפתחות של טכנולוגית המידע. התפתחות זו שיפרה את יכולת הארגון בעיבוד מידע ואפשרה לו להשתחרר

מהצורך להפריד את מערכת המידע לחלקים, כלומר, ללכת לקראת שילוב. ואכן, ההתקדמות הטכנולוגית הובילה לשילוב רב יותר של תת־מערכות המידע.

מערכת "משולבת" אינה זהה בהכרח למערכת "כוללת" (total information system). מערכת משולבת יכולה להיות מערכת כוללת או בלתי כוללת. בעוד שבמערכת כוללת הכוונה היא למערכת־על אחת, הרי מערכת משולבת יכולה להיות גם פדרציה או איחוד של תת־מערכות. הרעיון שמאחורי מערכת כוללת הוא רעיון של מערכת אחת מאוחדת, המטפלת בכל העיבודים של הארגון והמאחסנת את כל הנתונים במקום מרכזי אחד. הניסיון הוכיח, שהשילוב המושלם הגלום במערכת הכוללת הנו רעיון קשה מדי לביצוע ולהפעלה. הדעת הרווחת כיום, הן בספרות המקצועית והן בעקבות ניסיון מעשי שנרכש, מצדדת בחתירה לקראת איחוד של תת־מערכות הניתנות (כל אחת) לאחזקה ולפיתוח ברמה מסוימת של אי־תלות. שכן כאשר מדובר על איחוד ולא על מערכת אחת, עדיין נשארת מידה רבה של אי־תלות בכל מערכת שהיא תת־מערכת של האיחוד. חסידי המערכת הכוללת טוענים לשילוב מושלם של כל העיבודים. הניסיון מראה כי גישה זאת אינה מעשית. מצויים גורמים רבים מדי שיש לשקול אותם בו־זמנית, מה גם שאחזקת מערכת כוללת קשה ביותר. מסיבה זו גוברת הנטייה לגישה של פיתוח חוליות (מודולים) ראשיות, והשילוב נעשה רק כאשר הוא באמת נחוץ לצרכי הארגון, ולא עקב זאת שמנתח המערכות (או מומחה המחשבים) סובר שהשילוב יפה לארגון.

חשוב להבחין בין מערכת המידע לבין הארגון שהיא משרתת. בראש וראשונה על מערכת המידע לשרת את הארגון ואין היא כעין ישות בפני עצמה. גם הארגון וגם מערכת המידע הם, למעשה, מערכות המעורבות ביחסי תחלופה בין עצמאות לבין שילוב. ההנעה לשילוב חלקי מערכת המידע היא הגדלת התרומה לאפקטיביות של הארגון. שילוב מערכת המידע הוא תנאי הכרחי אך בלתי מספיק לשילוב פעילויות הארגון, כלומר, גם אם מערכת המידע תהיה משולבת ואפקטיבית, אין פירוש הדבר כי הפעילויות בארגון אשר משרתת מערכת המידע הן משולבות או אפקטיביות. התנאי המספיק הנדרש הוא התכנון. הארגון חייב לתכנן את עצמו להתחשב בכל המשתנים שבהם הוא תלוי ואת האינטראקציות בין הפעילויות השונות בתוך הארגון.

כדי לבנות מערכת מידע משולבת אפקטיבית, יש צורך, בראש ובראשונה, במודל אנליטי ובתאוריה משולבת של מערכות ארגוניות. אין עדיין תאוריה מגובשת על הנעשה בארגון, על הדרך בה נבנה הארגון ומה הם יחסי הגומלין בין המערכות השונות בתוך הארגון. לכן קיימים קשיים כה רבים בהקמת מערכת מידע משולבת המושתתת הן על ראייה נכונה של מודל הניהול והן על תורת הארגון.

עוד מראה הניסיון שקל יותר לבנות מערכת מידע משולבת לארגון בעל מטרה מרכזית אחת, שכן אז ניתן להוסיף לאותה מערכת מידע גם את המטרות המשניות של אותו ארגון. אולם, בארגונים עסקיים, החותרים בדרך כלל למספר מטרות,

קשה ביותר לבנות מערכת ממוחשבת שתתאים ליישום כל המטרות של הארגון. ואכן, המושג של מערכת מידע משולבת יושם בעיקר בארגונים בעלי מטרה מרכזית מוגדרת היטב, כאשר יתר המטרות, אם המדובר בניהול החשבונות או בניהול פיננסי, הן מטרות משניות וכאשר המטרה המרכזית מהווה, למעשה, את הישות המרכזית של אותו ארגון (למשל, חברות תעופה). היה, אפוא, קל יחסית לפתח מערכות מידע משולבות כאלה לארגוני תחבורה, למערכות צבאיות בעלות מטרה מרכזית אחת, או לבנקים ולבורסות לניירות ערך. לעומת זאת הניסיון הוכיח שבארגונים יצרניים-תעשיתיים, בעלי מגוון מוצרים נרחב ופעילויות נרחבות, קשה לבנות מערכת מידע משולבת.

יש לזכור, שכאשר מדובר במערכות מידע יש לארגון שני סוגים של מערכות מידע, פורמלית ובלתי פורמלית. גם קבוצה נכבדה מאד של מומחי מערכות אינה יכולה לבנות מערכת שתקיף את כל צרכי המידע הפורמלי והבלתי פורמלי בתוך הארגון, כיוון שאלו מסובכים ונרחבים במידה כזו שקשה למזג אותם באופן אפקטיבי. לפיכך, לגבי רוב הארגונים אי אפשר לדבר על מערכת מידע כוללת, מה גם שמערכת מידע בנויה מסביב למחשב המסוגל בעיקר לספק אותו מידע שניתן לתכנתו ולשלבו עם תאוריה, כלומר, מידע פורמלי. קשה מאד לבנות מערכת מידע שתהיה מבוססת על אספקת מידע בלתי פורמלי, איכותי, שהוא, למעשה, עיקר מעיינם של אנשי הדרג הניהולי הבכיר.

יש הטוענים להקמת מערכת מידע משולבת ואולי גם מערכת מידע כוללת שתקיף את כל צרכי הארגון. טענות אלו מבוססות בעיקר על ההצלחה המוקדמת של טכנולוגית המידע בטיפול במערכות בודדות, כמו, למשל, מערך שכר או מערך כוח-אדם. במקרים אלה היתה ההצלחה כה רבה, שיש נטייה טבעית אצל מנתחי מערכות להשתמש באותם כלים ובאותן שיטות לגבי מערכות המידע של הארגון כאשר רואים בו שלמות אחת. ברור, שאין זה הכרחי, שהצלחה בעבר תגרור הצלחה גם בעתיד, כיוון שהמשימות שונות לחלוטין. ולכן, המעבר מפיתוח מערכות בודדות שהצליחו אל פיתוח מערכת כוללת או משולבת עלול להיות מהיר מדי וחסר בסיס תיאורטי.

מערכת מידע משולבת, המשרתת את צרכי כל הארגון וכל דרגי הניהול הנה, אפוא, מערכת המאורגנת מחלקים שונים התומכים זה בזה. מערכת כזאת חייבת לספק מידע מספק עבור כל דרג ניהולי, מידע שחייב להיות רלוואנטי בזמן וכן עליו לענות על כל צרכי המידע של אותו דרג. ספק הדבר אם שילוב מושלם כזה ניתן להשגה מהר. שילוב אמיתי פירושו חתירה לקראת מערכת מידע מאוזנת, אשר זו מצדה נשענת, בהכרח, על חקירת צרכי המידע האמיתיים של הארגון על כל דרגיו.

כל ארגון מורכב מכיל, בדרך כלל, מספר יחידות המייצרות מוצרים או מספקות שירותים שונים ומבצעות פונקציות שונות. כל יחידה כזאת, העוסקת במוצרים ובשירותים שונים, נזקקת למידע משלה. מערכת הבקרה של הארגון חייבת לקלוט מכל אחד מאותם חלקים של הארגון את סוגי המידע, שמערכת הבקרה

הכללית נזקקת להם לצורך בקרה לתיאום בין אותם חלקים. פירושו של דבר שכל מערכת תפעולית בפני עצמה חייבת לייצר מידע למטרות מרכזיות. חלק מהנתונים המעובדים לצרכים פנימיים של אותו חלק בארגון חייב להיות מעובד בדרכים שונות לצורך מערכת הבקרה המרכזית. באופן דומה נזקקות המערכות התפעוליות השונות גם למידע על מערכות אחרות באותו ארגון וכן למידע על הארגון בכללותו. בעית השילוב היא להבטיח שתהליכי העיבוד במערכות התפעוליות ובמערכת המידע המרכזית, או מערכת הבקרה הכללית, יתמכו זה בזה באספקת נתונים ושהתיכון של מערכות אלו יעשה כך, שתהיה חלוקה יעילה של עיבוד הנתונים בין המערכות השונות — במלים אחרות, שתהיה חתירה לקראת אי־כפילות בין המערכות השונות.

בעית השילוב משמעותה גם השתקפות של השאלה הכללית יותר בניהול — ריכוז או ביזור. למשל, האם יעיל יותר שפעילות עיבוד המידע תבוצע בעיקר במערכות התפעוליות, עם פעילות מינימלית הנדרשת לתמיכה במערכת הבקרה המרכזית, או האם יהיה זה יעיל יותר שרוב המידע שבו משתמש הארגון יאוחסן ויעובד במרכז במסגרת מערכת מידע ניהולי מרוכזת. בשנים האחרונות, עם התפתחות טכנולוגית המידע, הפכה שאלה זו של ריכוז לעומת ביזור להיות ברת־יישום. למשל, השימוש בתקשורת מספק פתרונות ביזוריים גם לשאלת השילוב של מערכות מידע. הדגש בהשוואה בין ריכוז לביזור הוא בשימוש אפקטיבי במקורות אנושיים מוגבלים ובציוד יקר, המהווה יתרון כאשר הולכים לריכוז, מחד גיסא, ובגמישות ובפשטות של המטלות השונות, כאשר הן מושגות תמורת עלות מסוימת, מאידך גיסא. הפשטות והגמישות, בדרך כלל, מוזכרות כיתרון הטמון בביזור של מערכות מידע.

מעורבות ההנהלה חיונית ביותר לשילוב אמיתי, כדי להבטיח שמערכות המידע ותהליכי ההחלטות בארגון ישלימו זה את זה. דבר זה דורש, כמובן, מלבד מומחי מערכות בעלי רקע וחינוך מתאימים, גם מנהלים בדומה להם. יש לזכור שמערכת מידע אינה מטרה בפני עצמה, אלא כלי של הארגון. מערכות מידע וקבלת החלטות משולבות, המושתתות על מושגים, מודלים וחוקי החלטה ברורים, יעודדו בארגון שילוב של מושגים, של החלטות ואיחוד של מטרות. לגבי השילוב אין החתירה מכוונת לקראת מערכת מידע משולבת באופן טכנולוגי בלבד, אלא גם לקראת שילוב בפעילויות הארגון, כלומר לקראת שילוב כפול, הן של מערכת המידע והן של פעילויות הארגון. במלים אחרות, כדי לפתח מערכת מידע משולבת, חייבת הנהלת הארגון לפתח תחילה את פילוסופית השילוב שלה לגבי אותו ארגון. ללא זאת לא תהיה מערכת המידע המשולבת אפקטיבית בתוך אותו ארגון שבו היא הוקמה. לנושא של מערכות מידע משולבות יש, אפוא, שני היבטים: ההיבט הפילוסופי שהודגש לעיל וההיבט של הביצוע, כלומר, דרך הקמתן של מערכות מידע משולבות.

חשיבותו של המונח מערכת, כאשר מדובר במערכות מידע משולבות, היא בכך שהן האנשים והן הציוד וגם האלמנטים האחרים של המערכת קשורים זה בזה

כרכיבים של שלמות מתואמת או של שילוב. מערכת מידע גדלה בשילוב כאשר היא שוטפת בזרם מהיר יותר ובאופן אוטומטי, וכאשר המימשקים בין החלקים השונים של אותה מערכת הופכים יעילים ואפקטיביים. כל מערכת מידע בתוך ארגון חייבת להגיע גם לאותה רמת שילוב שהתנאים הכלכליים של אותו ארגון תובעים אותה — כלומר שיש להתחשב גם בעלות ולא רק בתועלת. מערכת מידע משולבת לארגון כולל ומסובך חייבת להיראות כמערכת־על של מערכות בודדות, או כאיחוד של תת־מערכות. זאת, כיוון שרק ארגון מונוליטי ביותר יכול להיתמך בידי מערכת כוללת אחת. איחוד זה דרוש לצורך אחידות במבנה, בתקנים, בתיעוד, בתהליכי הביצוע ובמטרות משותפות לארגון כולו וגם לצורך הקטנת העלות בהתקנת אותן תת־מערכות ובתפעולן. השילוב חייב להיבנות כמערכת של מערכות, כאשר כל מערכת בפני עצמה משולבת באופן פנימי בצורה איתנה וכאשר השילוב בין תת־המערכות הוא שילוב רופף. ככל שמגוון הפעילויות של הארגון רחב יותר, כן יהיה השילוב בין תת־המערכות רופף יותר. פירוש הדבר כי השילוב יהיה רופף יותר ככל שהארגון יהיה מסובך יותר ועם מגוון פעילויות רחב יותר, ובה בשעה כל תת־מערכת בפני עצמה משולבת באופן הדוק.

הגישה לפיה תהיה בנויה מערכת המידע כאיחוד של תת־מערכות ולא כמערכת אחת, כרוכה בהחלטה איסטרטגית הקודמת לתכנון, והיא ההחלטה על הגישה שבה יש לגשת אל נושא הקמת המערכת או תת־המערכות השונות. גישה אחת, אשר קראנו לה לעיל גישה של תקיפה "מלמעלה למטה", הנה גישה מונוליטית, גישה של פיתוח בבת אחת של מערכת מידע. גישה זו טובה (אולי) לארגון מונוליטי, כמו צבא, ארגון שיש לו הירארכיה קשיחה מאד של פיקוד וצורך בתגובה מתואמת, מהירה ומדויקת למצבים טקטיים קונקרטיים וקריטיים. אבל בארגונים אחרים, ואלה הם מרבית הארגונים העסקיים או הציבוריים, גישה זו יקרה מאד, מסוכנת מבחינה טכנולוגית, והבעיות המתעוררות בניהול המערכות המושתתות עליה הן עצומות. גישה אלטרנטיבית הנה הגישה האבולו־ציונית לתיכון, לניתוח ולהקמת המערכת. גישה זאת, עומדת, כמובן, בסתירה לאיסטרטגיה של תיכון כולל של כל מערכת־העל ואחר כך של ביצוע. הגישה האבולוציונית אין פירושה הגדרת צרה, אלא גם טווח של דרכים אבולוציוניות לתכנון ולהקמת מערכות מידע. בקצה אחד, הקרוב לגישה של מלמעלה למטה, עומדת ההחלטה להתפתח לכיוון מטרה ברורה ביותר, כלומר, המתכנן יודע לאן הוא שואף להגיע, אבל אין הוא יודע לבטח כיצד מגיעים. כאן הגישה האבולו־ציונית מתבטאת ב"איך" ולא ב"מה". המטרה במקרה זה ידועה ומוכתבת מראש, אולם הפיתוח נעשה בצורה אבולוציונית. בקצה השני של הגישות האבולוציוניות קיימת האבולוציה המקרית, שפירושה מין פיתוח אקראי ובלתי מבוקר שהוא די אופייני להרבה ארגונים.

ברור שאין מקום לשתי הגישות הקיצוניות. במקרה הראשון, ארגונים מודרניים הם מסובכים מכדי שיובנו ושתוגדר עבורם מטרה אחת ברורה מראש. ניתן להבין אותם ואת מטרותיהם ואת הפעילויות שלהם רק לאורך זמן או לאורך

תקופת למידה. כלומר, יתרון הלמידה כאן חשוב ביותר, כאשר מדובר בלמידה אגב תהליך הביצוע. התכנון חייב תמיד להיות מעודכן, איטרטיבי, כלומר, לחזור על שלבים שנעשו קודם מאחר שהתנאים נשתנו, או שנלמד דבר חדש. דבר זה נכון ביחוד לגבי ארגונים בעלי־קשת רחבה ביותר של פעילויות. תכנון ולמידה הולכים, אפוא, בד בבד. אנו לומדים איך לתכנן ובה בשעה מה לתכנן. לגבי הקצה הקיצוני השני של האבולוציה המקרית, איננו רוצים להגיע לערב־רב של יישומים כפולים וחופפים, אשר לא זו בלבד שהם מבזבזים מקורות ומשאבים, אלא גם מחמיצים את ההזדמנות הגדולה שלימוד רצוף ומתוכנן עשוי להקנות לארגון ולגלות לו את המטרות ואת הגישה הנכונה לאותם צרכי מידע שלהם נזקקות המטרות. הגישה לפיתוח מערכת מידע משולבת חייבת, אפוא, לעמוד באמצע, בין שני הקצוות הקיצוניים הללו: לדעת ככל האפשר לאן הפנים מועדות, ולהכיר בכך שככל שנרצה להתבונן קדימה כן תהיה הראייה פחות ממוקדת.

ההסכמה הכללית הרווחת היום היא, לפחות בתחום הספרות המקצועית ובתחום המעשי, שעד עתה היו הביצועים של מערכות מידע משולבות מאכזבים ולא הגשימו את הציפיות. עובדה היא כי מערכות עיבוד נתונים לא חדרו לדרגי הניהול הגבוהים, בעיקר עקב אי־הבנה יסודית של יחסי הגומלין בין הרמות של החלטות ניהוליות, המידע הנדרש להן והמקורות של המידע. התיאוריה שהיתה נפוצה עד היום אמרה שככל שיתפתחו מערכות לעיבוד עסקות (טרנסאקציות), כן הן תהפכנה לחלק ממערכת קבלת ההחלטות בדרגי הניהול הגבוהים. גישה זו גרסה כי יש להתחיל ברמה התפעולית הנמוכה יותר בהקמת מערכות לעיבוד עסקות, וככל שהמערכות הללו תלכנה ותגדלנה, ממילא הן תעלינה (אנכית) לרמות הניהוליות הבכירות יותר. תיאוריה זו קובעת, שכאשר המערכות התפעו־ליות תתרחבנה, ויפותחו בסיסי נתונים מרכזיים, הן תתפתחנה, לבסוף, למערכות מידע ניהוליות משולבות. הטעות שבתיאוריה זו נעוצה בכך, שהמידע הדרוש לקבלת החלטות בדרג הבכיר אינו נוצר במערכת לעיבוד העסקות. חלק ניכר מהמידע בכלל אינו נוצר בארגון, אלא בסביבה החיצונית. יש, אפוא, בכך משום נאיביות להאמין שהמערכות לעיבוד עסקות תתפתחנה בעתיד למערכות מידע ניהוליות.

לעומת זאת סוברת גישת ה"מלמעלה למטה" שאין אפילו לנסות בבניין מערכות תפעוליות לפני שניתן לתאר את המערכת הארגונית כמודל. כלומר אין לדבר על מערכות לעיבוד עסקות לפני שברור מהם המודלים הניהוליים של הארגון.

למעשה שתי הגישות גם יחד אינן ברות־ביצוע. הגישה, שנראית כברת ביצוע, דוגלת בפיתוח, **במקביל**, של שתי מערכות — מערכת עיבוד העסקות ומערכת המודלים הניהוליים. דבר זה נעשה, באופן קצר ופשטני ביותר, בצורה הבאה: בפיתוח מערכות מדברים אנו על שלבי מחזור־החיים של המערכת. המחזור הנו איטרטיבי, כלומר שבכל שלב מתבצעים הערכה ותכנון מחדש של המערכת לאור ההערכה. כאשר מבוצעים ההערכה והתכנון מחדש במחזור־החיים של כל

אחת משתי המערכות המקבילות, יש לנסות לשלב מערכת אחת עם רעותה. כלומר, בשלב ההערכה והתכנון מחדש של מערכת עיבוד העסקות חייבים לשלב את ההערכה של מערכת המודלים הניהוליים, ולהיפך — בעת הערכה והתכנון מחדש של מערכת המודלים הניהוליים יש תמיד לוודא מה שנעשה במערכת עיבוד העסקות. בצורה כזאת תיווצר במשך הזמן נטייה לשילוב מלא של שני סוגי המערכות, בחינת מטרה רצויה, אם כי רחוקה למדי, ואף לא אפשרית בהכרח.

IV. חלק רביעי

היבטים ניהוליים וכלכליים של מערכות מידע

1.

ארגון וניהול
יחידת עיבוד נתונים

ארגון יחידת עיבוד נתונים

מבנה יחידת עיבוד נתונים מבוסס על עקרונות ארגוניים המשותפים ליחידות ארגוניות רבות, אך נוספים עליהם עקרונות המיוחדים לארגון פעולת עיבוד הנתונים והנובעים מאופיה המיוחד של פעילות זו.

יחידת עיבוד נתונים צריכה, ככל ארגון, לקיים את הפונקציות הבאות:
- — הכרת הסביבה שבה פועלת היחידה ומטרותיה.
- — הגדרה ברורה של יחסי סמכות ודיוות.
- — הגדרה ברורה של קווי קומוניקציה בתוך היחידה ומחוצה לה.
- — איחוד פונקציות המביא לניצול כלכלי ויעיל של כוח־אדם וציוד.

למעשה מספקת יחידת עיבוד הנתונים שרות לשאר חלקי הארגון. לעתים מהווים אותו ציוד ואותם מתכננים ומתכנתים חלק ממערכות שונות בארגון, לפיכך, מטרת מערכת עיבוד נתונים טובה היא זו המביאה לשילוב של פעולות עיבוד הנתונים בין חלקיו השונים של הארגון. תכונות ייחודיות אלו גורמות להבלטת מספר תפיסות:
- — יש למקם את הפעילות לעיבוד הנתונים כך שתוכל לספק את צרכיהם של כל חלקי הארגון ביעילות
- — המבנה הארגוני צריך להיות מותאם לצורך בשינוי ובעדכונים מתמידים הנובעים מההתפתחות המהירה בענף המחשבים.
- — על המבנה הארגוני להבליט את פונקצית התכנון, התזמון והפיקוח כדי שהיחידה לעיבוד הנתונים תמלא את תפקידה.

— פונקצית עיבוד הנתונים צריכה להיות מאורגנת כך, שלא יהיה לשום פרט או יחידה קטנה שליטה מלאה על המידע המעובד. האחריות צריכה להתחלק כך, שמאמצי קבוצה אחת יבדקו את פעילותה של קבוצה אחרת.

גורמים סביבתיים

הבנת סביבתה של הפעילות לעיבוד הנתונים היא דבר בסיסי בעת ארגון הפונקציות השונות. בתהליך הארגון יש לפתור שתי שאלות בסיסיות:

— כיצד מתקשרת פונקצית עיבוד הנתונים עם הארגון כולו?
— מהן המטרות הייחודיות שעל פונקצית עיבוד הנתונים לשרת?

כדי לענות על שאלות אלו על המארגן להגיע לסיכומים הבאים:

— המצב והבעיות הנוכחיות של עיבוד הנתונים בארגון ומהן תוכניות הפיתוח לעתיד.
— מדיניות ההנהלה הבכירה בקשר למיקום עיבוד הנתונים והמטרות.
— ניתוח הדרך שבה משרתת יחידת עיבוד הנתונים את המשתמשים ומהי רמת השרות.

מטרות

הסיבות העיקריות לקיום פעילות עיבוד הנתונים בארגון הן המטרות. להלן דוגמא לכמה מטרות בסיסיות כאלו:

— תמיכה בפרויקט פיתוח אחד או יותר (למשל תכנון מוצרים, פיתוח תוכנה וכד').
— תמיכה בפעילויות תפעול (כגון פיקוח על הייצור, ניהול חשבונות עובר ושב בבנקים וכד').
— תמיכה בהנהלה (למשל מערכות מידע ניהולי, ניהול בקרה ותקציב, וכד').
— מתן שרותי מחשב למשתמשים במידת הצורך.

כמובן שפעילות מסוימת יכולה לשרת יותר ממטרה אחת. אולם, המטרה העיקרית היא זו המשפיעה בעיקר על המבנה הפנימי של הפעילות.

גורמים אחרים

גורמים סביבתיים אחרים הם:

— מספר וגודל המשתמשים בעיבוד נתונים: האם קיים מספר גדול של משתמשים קטנים, או מספר קטן של משתמשים גדולים? כיצד מתחלק עומס העבודה ביניהם? וכד'.
— תלות ההנהלה בשרותים הניתנים על-ידי עיבוד הנתונים: האם פלט עיבוד הנתונים והקבצים הם דבר חשוב ונדרש להנהלה לתכנון? האם דרושה להנהלה גישה מהירה למידע?

— תלות החלק התפעולי בשרותים הניתנים על־ידי עיבוד הנתונים: האם עיבוד הנתונים הוא חלק אינטגרלי של העבודה המבוצעת? האם זמינות מהירה של המידע היא גורם חשוב בתפעול השוטף?
— סוג עיבוד הנתונים: האם הוא מבוסס בעיקר על מסדי נתונים? עיבודים מדעיים? פיקוח על תהליכים? וכד׳.
— איזה חלק מהתכנון והתכנות נעשה על ידי המשתמשים עצמם? האם פונקצית עיבוד הנתונים פועלת כלשכת שרות? האם מקיימת פונקצית עיבוד הנתונים מאגר של אנשי מקצוע? האם כל עבודת הפיתוח נעשית על ידי פונקצית עיבוד הנתונים?
— היחס שבין עבודות פיתוח לבין עבודות אחזקה: האם מרבית המשאבים מוקדשים לעבודות פיתוח או לעבודות אחזקה ועדכונים?
— תחזיות של עומס עבודה ולוח זמנים: האם ניתן בקלות לתזמן עבודות עיבוד נתונים? האם ניתן לבצע את העבודות בזמן סבב קצר ללא הודעה מראש?
— סוגי הקלט למערכות עיבוד הנתונים: בעזרת מסופים או כרטיסים? האם יש צורך להפוך את נתוני הקלט למצב שבו ניתן לקלטם? וכד׳.

סביבות טיפוסיות

מספר הצורות בארגון הסביבתי הוא כמספר הקומבינציות של גורמי הסביבה. בחרנו בארבע דוגמאות כדי להמחיש כיצד פרופילים סביבתיים שונים זה מזה. לאחר מכן נראה כיצד משפיעים הפרופילים הללו על צורת הארגון של יחידת עיבוד הנתונים (ראה טבלה 4.1).

הפעילות הטיפוסית הראשונה אופיינית בדרך כלל ליחידת עיבוד נתונים בגודל קטן עד בינוני, יחידה המשרתת את ההנהלה, את מדור הכספים, משכורת, רכש וניהול המלאי. הקבצים מעודכנים בזמנים קבועים וכן מיוצר פלט בזמנים קבועים. בדרך כלל אין קיימים מסופים, רוב הנתונים מיוצרים מטופסי מקור המשתמשים תלויים לגמרי בפעילות היחידה. פיתוח המערכות ברובו כבר נעשה, ופעילות המתכננים והמתכנתים קשורה בעיקר באחזקה ובתיקונים קטנים.

הפעילות הטיפוסית השנייה עיקרה מחקר ופיתוח. בעניין זה הדרישה לפיתוח ותכנות מערכות גדולה ביותר והדבר נעשה בדרך כלל בשיתוף עם המשתמשים. עומס העבודה הוא גדול למדי ובלתי צפוי. מספר קטן של משתמשים עומד על כך שיקבל עדיפות שווה וזמן סבב מידי. בדרך כלל המחשב יהיה בינוני עד גדול והקלט מגיע בצורות שונות ומשונות. לעתים הוא בלתי אמין.

הפעילות הטיפוסית השלישית היא זו הנותנת שרותי מחשב, אם על־ידי שרותים פנימיים או על־ידי לשכת שרות מסחרית. מטרתה העיקרית מתן שרותי חישוב כדרישת המשתמשים למספר גדול של יישומים ומטרות. קיים מספר גדול של משתמשים, חלק מהם קשור למחשב על־ידי מסופים מסוגים שונים והחלק האחר מעבד את יישומיו בעיבוד בסדרות. ניתן לצפות את עומס העבודה אך לא על

בסיס של עבודות ספציפיות. כמובן, שהמשתמשים מבצעים את כל התכנון והתכנות בעזרת אנשי יחידת עיבוד הנתונים כדי להגיע לשימוש יעיל יותר של הציוד.

הפעילות הרביעית היא זו המשתמשת בקשר ישיר למסדי הנתונים. מסדי הנתונים הם החלק העיקרי והחשוב של הפעילות לעיבוד הנתונים ולמעשה חלק אינטגרלי של פעילות הארגון. מסדי הנתונים הנם גדולים, ומעודכנים באופן שוטף על־ידי קשר עם מסופים המספקים את נתוני הקלט. שאילתות רבות מוזרמות, כמובן, למסדי הנתונים וניתן לחזות בדרך כלל את העומס הצפוי. יישומים חדשים מתווספים ללא הרף כאשר פיתוח המערכות נעשה בצורה מרכזית על־ידי יחידת עיבוד הנתונים. במקביל נעשית אחזקה שוטפת של המערכות הקיימות ולעתים מבוצעים שינויים קלים במערכות אלו.

סוגי פעילויות של עיבוד נתונים

פעילות גורמים	תמיכה	מחקר ופיתוח	שרותי מחשב	תמיכה במסדי נתונים בצורה ישירה (on-line)
מטרות	תמיכה בהנהלה	תמיכה בפרויקטים	כוח חישוב עבור משתמשים	תמיכה בתפעול
המספר והאיזון של המשתמשים	מספר נמוך איזון שווה	נמוך ; ביקוש גדול ; איזון שווה	גבוה, לא מאוזן	נמוך ; ביקוש שווה
תלות ההנהלה ביחידת עיבוד הנתונים	חשובה, אבל לא בזמן אמיתי	נמוכה	גבוהה	בינונית
תלות התפעול	חשובה, אבל לא בזמן אמיתי	גבוהה מאד, הפעילות חלק מהפרויקט	בינונית לגבוהה	גבוהה, הפעילות חלק מהתפעול
סוג עיבוד הנתונים	עיבוד בסדרות ; רציף	עיבוד בסדרות ושיתוף זמנים ; הרבה חישובים	תערובת של עיבוד בסדרות ושיתוף זמנים	שימוש גדול במסדי נתונים ; מספר רב של נתונים, שיתוף זמנים
תלות המשתמשים בפיתוח מערכות	לא נעשה פיתוח על־ידי משתמשים	משתמש ויחידת עיבוד הנתונים	הפיתוח כולו על־ידי המשתמשים, להוציא תכנות מערכות	רוב הפיתוח על־ידי מחלקת עיבוד הנתונים
פיתוח ואחזקה של מערכות	מעט פיתוח ואחזקה	פיתוח רב ומעט אחזקה	פיתוח מועט ואחזקה רבה	פיתוח ואחזקה גדולים
חיזוי עומס העבודה	ניתן לחיזוי	קשה להיזוי	ניתן לחיזוי בצורה סטטיסטית בלבד	אפשרות חיזוי בינונית
קלט	נתונים בסיסיים	בדרך כלל נתונים בסיסיים ; לפעמים בצורה ישירה	ניתן לעיבוד מידי	על־ידי מסופים ; נתונים בסיסיים וכאלו שניתנים לעיבוד מידי

מיקום יחידת עיבוד הנתונים במסגרת הארגון

קיימות שלוש אלטרנטיבות למיקום יחידת עיבוד הנתונים במסגרת הארגון:

מיקום תפעולי

לפי תרשים 4.1, שלוש פעילויות של עיבוד נתונים מתקיימות באופן עצמאי במסגרת הארגון ומשרתות את פונקציות המימון, ההנדסה והייצור. שלושת הפונקציות המרכזיות האחרות, מחקר ופיתוח, שיווק ומינהל יכולות לשכור שרותים באחת או יותר מהפעילויות בארגון או מחוצה לו.

מבנה כזה מתאים במיוחד לארגון מבוזר, בדרך כלל ארגון גדול שאין בו שילוב של בסיס נתונים. מבנה כזה ייתכן גם כתוצאה מפיזור גיאוגראפי, או מקיום פונקציות שונות לחלוטין באגפים השונים. העברת מידע בין הפעילויות העצמאיות תיתכן אז באמצעות מערכת תקשורת, העברת סרטים, או על ידי העברה ידנית.

יתרונות המבנה:

א. פיקוח צמוד של הפונקציות הארגוניות על יחידת עיבוד הנתונים שבשליטתם.

ב. כל יחידה יכולה לרכוש רק את הציוד הדרוש לה ללא הזדקקות ליחידה גדולה הדורשת התאמות לעבודות השונות.

ג. מתאים לגישה של עלות ורווח לפי מרכזי רווח.

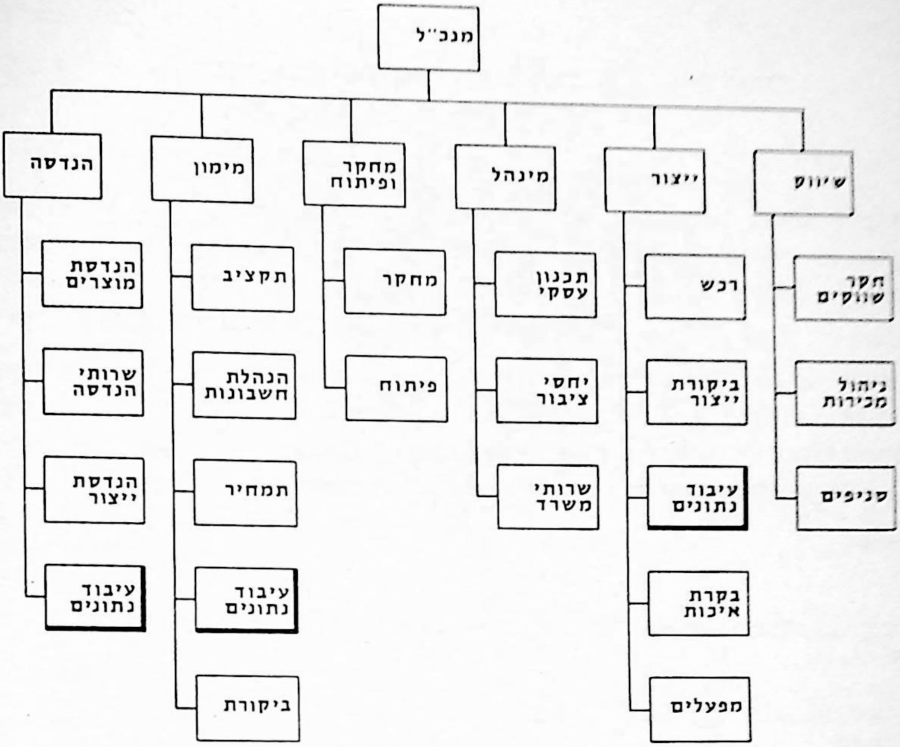
חסרונות המבנה:

א. ייתכן בזבוז מסוים של ציוד וכוח־אדם מתוך כפילות מסוימת בתפקידים.

ב. קיימים קשיים בשילוב של בסיסי נתונים.

ג. קביעת תקנים ושליטה על תהליך עיבוד הנתונים קשות יותר, משום שהיחידות כפופות לפונקציות ארגוניות ולא למרכז לעיבוד נתונים.

וריאציות שונות לשיטה זו; המשותף להן בדרך כלל הוא ניסיון לקשר את פעילויות עיבוד הנתונים העצמאיות באמצעות תפקיד "החותך" את הקווים הארגוניים בתרשים.



תרשים 4.1: תרשים ארגוני — עיבוד נתונים הנו חלק מהיחידות התפעוליות.

מיקום במסגרת אגף המינהל

בתרשים 4.2 נמצאת יחידת עיבוד הנתונים כמחלקה שוות-מעמד באגף המינהל, המספק שרותים מרוכזים לכל יחידות הארגון. אם יש צורך במיקום הציוד והמשאבים במקומות מרוחקים (מפוזרים מבחינה גיאוגראפית), או בפונקציות שונות בארגון, נשאר הפיקוח והניהול בידי יחידת עיבוד הנתונים המרכזית.

בגישה זו "משלמות" המחלקות השונות עבור שרותי המחשב לפי קביעת מחירים פנימית של הארגון. היא מותאמת לארגונים בינוניים וקטנים וכן לארגונים גדולים שבהם קיים צורך בשילוב רב של בסיסי הנתונים.

יתרונות הגישה:

א. כל המחלקות מקבלות טיפול שווה.
ב. כל המשתמשים יכולים ליהנות מיכולת חישובית גדולה יותר, עקב הימצאות מחשב גדול יותר.
ג. שליטה נוחה יותר על מחלקת עיבוד הנתונים ובתוכה.
ד. אפשרות חסכון בכוח-אדם, בעיקר בדרגי הנהלת היחידה.

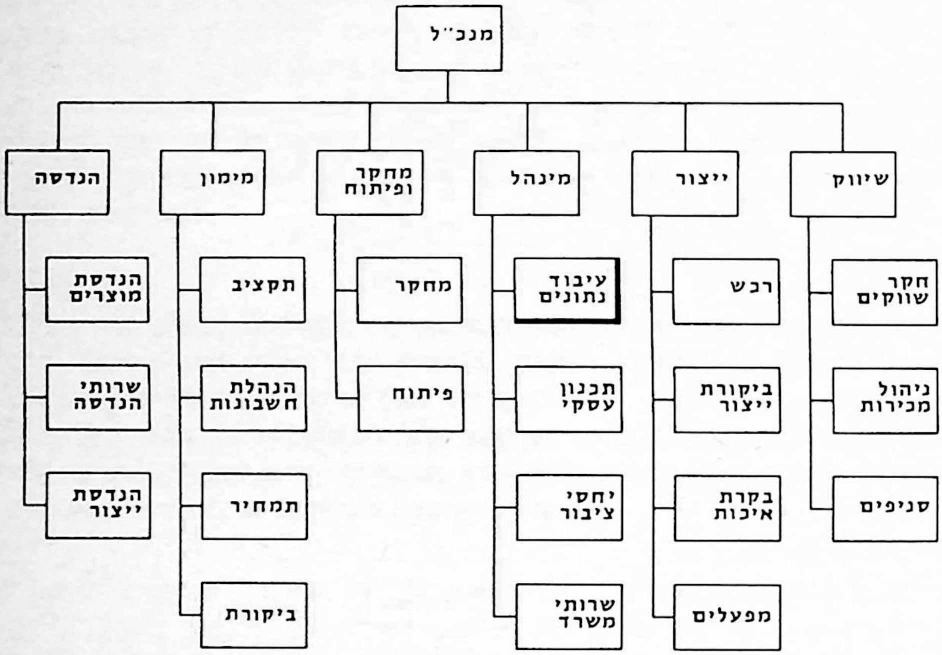
ה. הרחבת פעילותו של משתמש אפשרית במסגרת היחידה ואין צורך בהרחבת ציוד לשם כך.

חסרונות הגישה:

א. המתכננים והמתכנתים אינם מכירים מקרוב את הפונקציות השונות בארגון.

ב. יש צורך בתמחיר מדויק של שרותי המחשב, אחרת נוצרת נטיה "לבזבז" את שרותי המחשב.

ג. קיים קושי של העדפות והקצאות בין המשתמשים השונים.



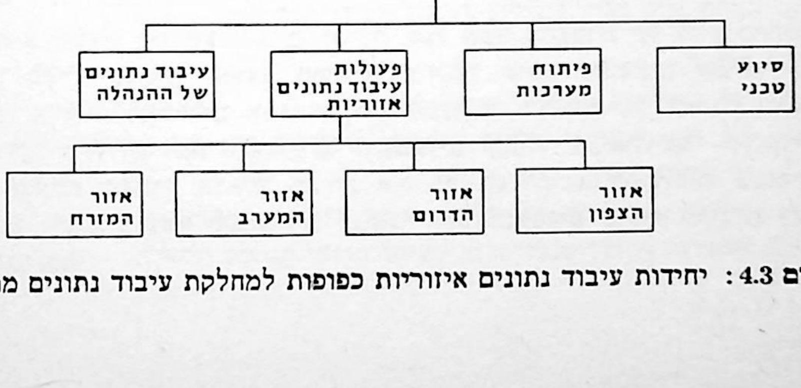
תרשים 4.2: תרשים ארגוני — עיבוד נתונים הנו חלק משרותי אגף המינהל

תרשים 4.3 מציג ואריאציה אחת לגישה זאת אשר מתגברת על חלק מהחסרונות שצוינו לעיל. במקרה המוצג מפעילה מחלקת עיבוד הנתונים חמישה מרכזי חישוב: ארבעה במשרדים איזוריים ואחר במשרד הראשי של הארגון. המרכזים האיזוריים משרתים את המשרדים והמפעלים באזור הגיאוגראפי שבאחריותם, ואילו המרכז במשרד הראשי משרת את כל הפעילויות המרוכזות במשרד זה וכמו כן יישומים אשר משתמשים בנתוני קלט הנאספים ברמה מקומית ושאינם מעובדים שם.

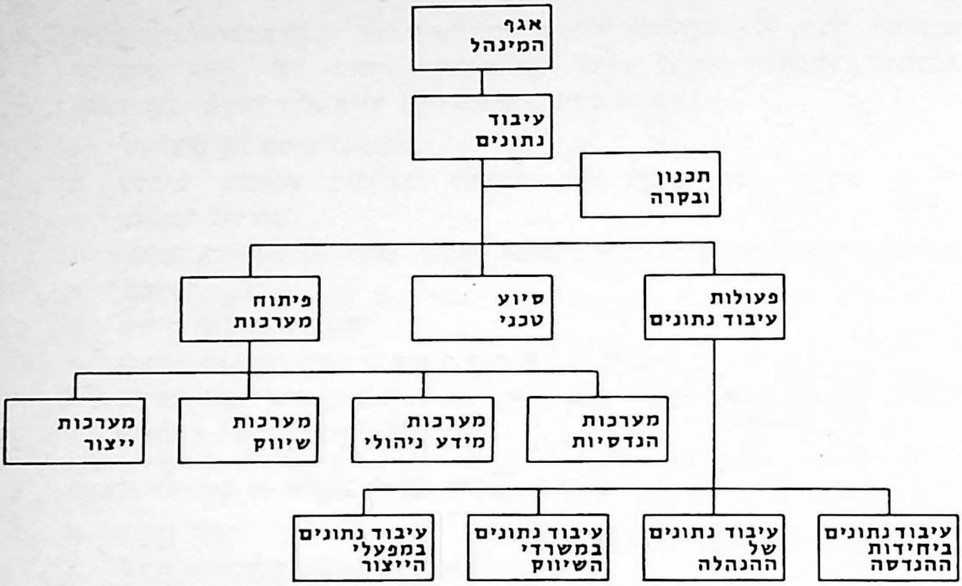
כוח האדם במשרדים האיזוריים הוא חלק אינטגרלי של מחלקת עיבוד הנתונים המרכזית, וכל משרד איזורי כפוף למרכז. פיתוח ותכנות יישומים סטנדרטיים נעשה במרכז, אולם ייתכן שמנתחי מערכות ומתכנתים יימצאו במשרדים האיזו־ריים כדי לבצע עבודות הקשורות אך ורק למשרד האיזורי.
הרכב הציוד הוא שונה, מותאם לכל מרכז איזורי לפי הצרכים, וכן גם עם הציוד הנמצא במרכז. ניתן כמובן לקשור את המרכזים השונים עם המטה במרכז על־ידי שימוש במערכות תקשורת.
תרשים 4.4 מציג ואריאציה אחרת לגישה הנ״ל, אולם במקרה זה נמצאות תת־המחלקות של עיבוד הנתונים במחלקות התפעוליות עצמן. פיתוח המערכות נעשה במקרה זה על־ידי התמחות בארבעה יישומים עיקריים: ייצור, שיווק, ארגון והנדסה. קבוצות אלו פועלות על מערכות סטנדרטיות לשימוש באמצעות מחלקות התפעול. גם במקרה זה פועל כוח האדם ברמה המקומית אולם הפיקוח והניהול נעשה על־ידי מחלקת עיבוד הנתונים המרכזית. גישה זאת מאפשרת התמחות בשטחים שונים, אך משאירה את הפיקוח והניהול בידי מחלקת עיבוד הנתונים המרכזית.

מיקום עצמאי

בגישה זאת נמצאת מחלקת עיבוד הנתונים תחת פיקוח ישיר של מנהל הארגון כולו. למחלקה זאת ניתנה אותה חשיבות, עצמאות ותשומת לב ההנהלה, דוגמת מחלקות עיקריות אחרות בארגון (כמו שיווק, מימון).
גישה זאת דומה לגישה הקודמת שבה ממוקמת מחלקת עיבוד הנתונים במסגרת אגף המינהל, ותתכנה אותן ואריאציות של יחידות עיבוד נתונים כפופות באיזורים גיאוגראפיים שונים, או במחלקות תפעוליות שונות.



תרשים 4.3: יחידות עיבוד נתונים איזוריות כפופות למחלקת עיבוד נתונים מרכזית



תרשים 4.4: יחידות עיבוד נתונים כפופות למחלקות התפעוליות

פונקציות בעיבוד נתונים

בכל יחידת עיבוד נתונים צריכות להתבצע מספר פונקציות. צורת קיומן ותרגומן לתפקידים ספציפיים שונה מיחידה ליחידה בהתאם לסביבה שבה היא פועלת.

פונקצית פיתוח מערכות

מערכת היא קבוצת תוכניות ונהלים המופעלים על הקלטים והקבצים הקיימים כדי ליצור פלט מבוקש. כל מערכת חייבת, כפי שנאמר, לעבור ניתוח, תיכון, תכנות, ניסוי וכן צריך לתחזק אותה ולבחון אותה באופן מתמיד. להלן הפירוט המתבקש:

1. ניתוח מערכות — מטרתה של פעילות זו מתן פתרונות טובים ביותר לצרכים נתונים, בהתחשב במטרות ההנהלה ובמשאבי הארגון (ציוד וכוח־אדם). בפעילות זו כלולים:

א. בחירת המערכות שיש לפתחן.

ב. ניתוח פעילויות קיימות ומתוכננות כדי לשפרן; איסוף מידע בקשר לפעילויות אלו וארגונו בצורה משמעותית לניתוח (תרשימי זרימה, גרפים וכו׳).

ג. ניתוח עלות/תועלת של גישות אלטרנטיביות לתכנון המערכת.

ד. ניתוח צרכיהם של בסיסי נתונים ופיתוח בסיסי נתונים המותאמים לצרכים הספציפיים הקיימים.

מערכות פירושו — תיכון מדויק של המערכת על סמך הצרכים
. הפלט של תיכון מערכות הוא מידע מדויק המאפשר למתכנת
ת תפקידו ללא צורך במידע נוסף. מידע זה כולל:
רטים של בסיס הנתונים.
כיבי מערכת מהולקים לשלבים אופרטיבים, סדר שלבים מוגדר
קשר ביניהם.
רה מדויקת של צרכי עיבוד הנתונים בכל שלב, כולל ציוד ומערכת
עלה.
רש שיטות התכנות.
רה הקובעת דרכי רישום של תהליכים ונהלים.
רט התהליכים שיתבצעו אם יהיה צורך לעדכן את המערכת בשלבי
תכנות, הניסוי או היישום.

יישומים — תכנות השלבים השונים במערכת:
בון לוגי.
וד ההוראות בשפה שנבחרה ליישום.
גום (קומפילציה) וניסוי התכנות.
צוד מדויק של התוכניות וההוראות להפעלתן לפי הנהלים הקיימים
חידה.

מערכת:
ון שיטות ניסוי לבדיקת המערכת לפי התקנים שנקבעו.
רת נתוני ניסוי המייצגים נכונה את התנאים שבהם תפעל המערכת.
המערכת, עדכונה ותיקון ליקויים.

המערכת — הכנסת מערכת חדשה לפעולה:
ן שלבי המעבר למערכת חדשה.
ן טפסים, הוראות וחומר לימודי למשתמשים.
ת המשתמשים ואנשי עיבוד הנתונים במערכת החדשה.
ב של המערכת החדש העם המערכות הקיימות.

מערכת:
קטנים במערכת ההפעלה הנובעים מהפעלה שוטפת. הפעולות
הן למעשה תמצית הפעולות בסעיפים 1—5.

ועי המערכת:
פעם יש לבדוק האם המערכת עומדת במטרות ובתקנים שנקבעו
התכונות האופרטיביות — זמני הפעלה, אחוזי שגיאה וכו׳.

ג. מידת ההתאמה למערכות אחרות ואפשרויות שילוב עם מערכות חדשות.

ד. מידת נצולת הציוד.

ה. מידת יעילות הנהלים.

פונקציות אופרטיביות

הפונקציות האופרטיביות בעיבוד נתונים אנלוגיות לאלו בכל מפעל יצרני אחר:

1. פיקוח ותזמון — רוב יחידות עיבוד הנתונים עומדות לפני ביקוש משני סוגים:

א. מתוכנן.

ב. בלתי מתוכנן.

על היחידה לתזמן את העבודות בצורה המביאה ליעילות מירבית. עבודה זו כוללת:

א. תכנון עבור עבודות מתוכננות.

ב. פיקוח ותיאום יומיומיים המביאים בחשבון קלקולים בציוד, עבודות בלתי צפויות וכו׳.

ג. פיקוח על זרימת נתוני הקלט והפצת הפלטים.

ד. קשר עם המשתמשים לגבי דרישותיהם ומצב עבודותיהם.

2. הפעלת הציוד — כל הכרוך בהפעלה שוטפת של ציוד עיבוד הנתונים, הכולל:

א. ציוד להכנת הקלט.

ב. ציוד הפיקוח של מערכת התקשורת.

ג. ציוד אחר, הכולל לדוגמא ממיינות, מפרידות נייר וכד׳.

ד. הפעלה ופיקוח של ציוד המחשב, מסופים, ציוד היקפי המחובר ישירות למחשב, הפעלת קוראות הכרטיסים, מדפסות וכד׳.

3. פונקצית סיוע להפעלה:

א. תחזוקה שוטפת של ספריית סרטים ותקליטים.

ב. תחזוקה שוטפת של ספריית תיעוד הפעלה.

ג. ניהול שוטף של מלאי חומרים הדרושים בתהליך.

ד. איסוף נתונים ודיווח על ביצועי המערכת.

ה. הגנה פיסית על הציוד.

סיוע טכני

1. פיתוח תקנים לעיבוד נתונים — קביעה, פרסום והשגחה על ביצוע כל

הפונקציות בדרכים, ולפי התקנים שנקבעו. תקנים אלה קובעים את:

א. שיטת העבודה.

ב. הרישום הנדרש בכל פעולה.

ג. מגבלות.

ד. מבנה קבצים, גודל הקבצים, בטיחות הקבצים וכד׳.

ה. ביצועים מינימליים הדרושים מכ״א וציוד.

2. ייעוץ טכני:

א. ייעוץ להנהלה ולמשתמשים בתכנון ובניצול היכולת של יחידת עיבוד הנתונים.

ב. ייעוץ לפתרון בעיות שוטפות ושאלות של המשתמשים.

ג. קיום ספרייה טכנית לשימוש המשתמשים.

ד. פרסום חוזרים, מודעות ועדכונים במערכת.

ה. פיתוח קורסי שרות.

3. תכנות המערכת:

א. בחירת ויצירת מערכת הפעלה.

ב. קביעת תקנים לגבי שימוש בתוכנה כללית ותוכניות עזר.

ג. סיוע וייעוץ למתכנתים בשימוש בתוכנה כללית ותוכניות עזר.

4. ניהול בסיס נתונים:

א. תכנון מבנה בסיס נתונים המותאם לצרכיה של היחידה והמתחשב גם בצרכים צפויים בעתיד.

ב. קביעת תקנים לשימוש בבסיס הנתונים.

ג. קביעת נהלי בטיחות לגבי שמירת בסיס הנתונים.

ד. עדכון סעיפים א, ב, ג בהתאם להתפתחויות שוטפות.

ה. אחזקת מילון לבסיס הנתונים.

5. ניהול מערכת תקשורת:

פיתוח, תכנון ופיקוח של מערכות תקשורת. פעילות זו כוללת:

א. קביעת התכונות הנדרשות מיחידה במערכת.

ב. סינון האלטרנטיביות לציוד בהתאם לדרישות ותכונות ורכישת הציוד המתאים.

ג. ייעוץ והדרכת המשתמשים השונים בציוד ובנהלים.

ד. קביעת התקנים לשימוש ביחידות תקשורת.

ה. תחזוקה.

מינהלה והנהלה

הנהלת יחידת עיבוד נתונים כוללת את כל פונקציות הניהול הקיימות בכל ארגון העוסק בפעילות מורכבת ומגוונת. מודגשות במיוחד הפונקציות הבאות:

1. פיקוח וניהול פרוייקטים: לעתים תמצאנה שתי שרשראות פיקוח ביחידת עיבוד נתונים: האחת רגילה כבכל ארגון, והשנייה הכרוכה בניהול פרוייקט שמאורגן לרוחב הקווים הארגוניים, דהיינו, לא במסגרת פונקציה מסויימת בארגון.

2. תכנון: הערכת הדרישות העתידיות והמקורות הדרושים לסיפוק צרכים אלו. עקב ההתפתחות המהירה בענף המחשבים והמדע הכרוך בו יש צורך בהגדרה ברורה של טווחי התכנון: קצר, בינוני וארוך. כן צריכים לעדכן באופן שוטף את המתכננים השונים בארגון ואת הלקוחות בארגון.

3. ביטחון:
   א. ביטחון פיזי הכולל בחירת מבנה מתאים.
   ב. נהלים והקצאת סמכויות ותחומי אחריות.
   ג. שמירה על קבצים.

4. דיווח:
   בעית "תרגום" המונחים הטכניים לשפה המובנת לכל אנשי הארגון.

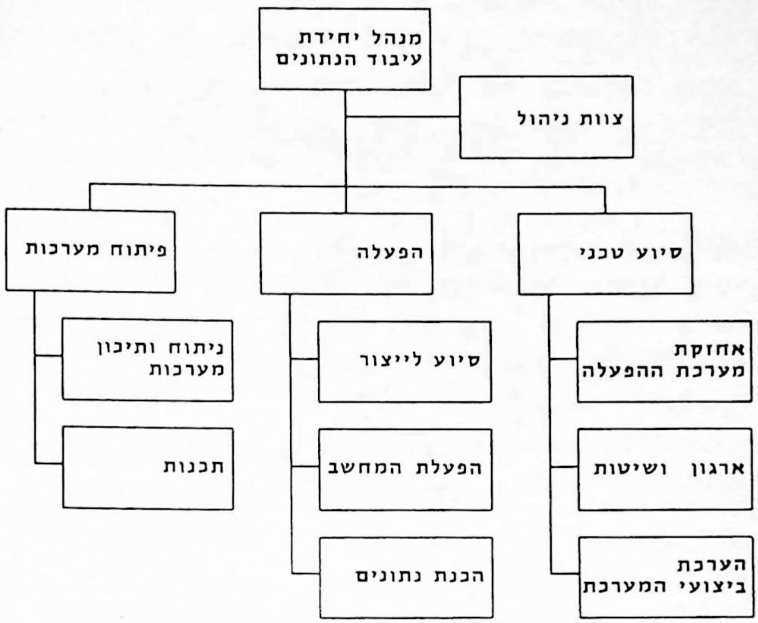
5. ניהול כוח־אדם —
   א. הגדרת התפקידים, סמכויותיהם ותחום עיסוקם.
   ב. גיוס ובחירת כוח־אדם מתאים.
   ג. הדרכת כוח־אדם חדש וקיום קורסים לכוח־אדם קיים.

6. ניהול פיננסי:
   הקצאת עלות המחשב בין לקוחותיו, תכנון פיננסי, רכישת ציוד וכו׳.

7. מינהל:
   נתינת שרותים כלליים הנדרשים בכל פעילות ארגונית:
   א. שרותי מזכירות.
   ב. רישום ותיוק.
   ג. אחזקה ונקיון.

דוגמא למבנה ארגוני בסיסי של יחידת עיבוד נתונים מוצגת בתרשים 4.5 במבנה כזה מחולקת הפעילות לשלושה חלקים תפעוליים ראשיים וקבוצת מינהלה אחת. כמובן שכל קבוצה מתחלקת בתוכה לפי הצורך.

ביחידות לעיבוד נתונים קטנות יותר, ניתן במידת הצורך לאחד את קבוצות פיתוח המערכות והסיוע הטכני לקבוצה אחת.



תרשים 4.5: מבנה ארגוני בסיסי של יחידת עיבוד נתונים

תיאור תפקידים עיקריים במסגרת יחידת עיבוד הנתונים

מנהל עיבוד הנתונים

מתכנן ומפקח על כל הפעילויות לעיבוד הנתונים בארגון. מתכנן שיפורים בפעילויות הארגון בעזרת מערכות חדשות או משופרות. אחראי על קיום קשר עם המשתמשים ותמיכה בשיפור פעילויות הארגון, שיפור השיטות וניצולת טובה יותר של המשאבים. מארגן את משאבי עיבוד הנתונים כדי לתת שרותים יעילים ותועלתיים למשתמשים.

תמחירן ותקציבן מחלקת עיבוד הנתונים

מתאם את תקציבי פעילויות עיבוד הנתונים. מיישם מערכת נהלים אשר מכינה הערכות תקציב ומדוות על הוצאות. מפקח על התחייבויות והוצאות בהשוואה לתקציב. מכין את תקציב עיבוד הנתונים עבור מערך הכספים של הארגון. מכין שיטות תמחיר ומנהל את מערכת החיוב עבור השימוש במשאבי מחלקת עיבוד הנתונים.

מנהל ניתוח ותיכון מערכות

אחראי על פרוייקטים לפיתוח מערכות. מתכנן ומפקח על עבודתם של מנתחי ומתכנני המערכות. מתכנן את המשאבים בצורה שתספק את הדרישות והאיכות של הפרוייקטים לפיתוח מערכות. מבקר את תיעוד המערכות, שלמותן, איכותן, והתאמתן לסטנדרטים.

מנתח מערכות

אוסף ומנתח מידע הקשור בפיתוח ועדכון מערכות עיבוד נתונים. מתכנן ומגדיר מערכות ושיטות ליישומן. מפקח או מנחה את יישומן של המערכות. מנתח מערכות פועלות וממליץ על שיפורים. פועל בשיתוף עם המשתמשים לשם איסוף מידע והגדרת מטרות המערכות. מתעד את הממצאים ואת תוצאות הבחינה. מציג את הממצאים, ההמלצות והדרישות בצורה פורמלית.

מתכנן מערכות

מתרגם את הדרישות אשר הוגדרו על־ידי מנתח המערכות לתכנון לוגי, כלכלי ומעשי. מכין דרישות מפורטות שלפיהן ייכתבו התוכניות הדרושות. דרישות אלו כוללות תרשימי זרימה אשר מציינים קשר בין תת־המערכות, זרימת הנתונים ומשאבים דרושים. מכין תיעוד לנהלי טיפול בנתוני הקלט, לעיבודים ומתכנן הפלט לפונקציות השונות.

מנהל תכנות

מתכנן, מפקח ובוחן את עבודת אנשי התכנות בפיתוח תוכנה חדשה או בעדכון תוכנה קיימת. מתכנן ומבצע הכנסת שיטות סטנדרטיות לארגון, קידוד ובחינת תוכניות. מקיים מעקב אחר תוכניות שהושלמו וכאלו אשר נמצאות בפיתוח בכל הקשור לשלמות, יעילות, דיוק ועמידה בסטנדרטים.

מתכנת

מתכנן ובודק תוכניות, כותב תוכניות ומכינן להפעלה על המחשב. מכין תרשימי זרימה של התוכניות הדרושות לעיבוד נתונים. עוזר למתכנן המערכות להכין דרישות לקבצים והגדרות לעיבודים. מבצע תכנות לפי סטנדרטים קבועים. בודק ביסודיות את פעולת התוכניות המושלמות בזיקה לתוכניות אחרות.

מנהל הפעלה

אחראי על כל הפעילויות הקשורות בביצוע עיבוד הנתונים במחשב. אחראי על פעולת ציוד מערכת המחשב. מכין נהלים הקשורים בביצוע העבודה להבטחת איכות וביטחון. מנתח שינויים בעומס העבודה ועוקב אחר יישומים חדשים אשר נמצאים בפיתוח כדי לחזות את ההשלכות על התפעול השוטף.

.2
הערכת מערכת תוכנה/חומרה חדשה

תהליך הערכת מערכת שונה מארגון לארגון בהתאם למידת הניסיון של הארגון בשטח המחשבים, הדחיפות בהפעלת המערכת הנבחנת וגורמים אחרים.

גישה נכונה תכלול בדרך כלל את השלבים הבאים:

1. בדיקת כדאיות (feasibility study).
2. הכנת מפרט.
3. קבלת הצעות.
4. בחינת הצעות.

ההשגחה והפיקוח על ביצוע השלבים נתונים בידי ועדה, או **קבוצת עבודה**, הכוללת בדרך כלל מנהלי דרג ביניים אשר מייצגים את הפונקציות השונות בארגון וכן נציג של המערכת הנבחרת.
רצוי שלחברי הקבוצה יהיה ידע בסיסי בעיבוד נתונים. נוסף על כך יש להסתייע בכוח־אדם מקצועי מהארגון, במידה וכבר יש יחידת עיבוד נתונים, ובאמצעות ייעוץ חיצוני במידה ואין. תפקידם של יועצים מבחוץ הוא להנחות, להסביר, להדריך ולסייע לחברי הקבוצה בתחומים טכניים אשר הקבוצה אינה בקיאה בהם. לא מומלץ בדרך כלל להפקיד את כל תהליך ההערכה בידי יועצים מן החוץ.

בדיקת כדאיות

בדיקת הכדאיות מטרתה לחקור את המערכת הקיימת, להעריך את הצורך בהכנסת או החלפת מערכת ציוד ותוכנה (hardware/software) ולבחון מערכות שונות מבחינות העלות, היעילות וההשפעה על הארגון.
כדי לבחון נכונה את הצורך במערכת חדשה, על קבוצת העבודה להכיר היטב את דרכי עיבוד הנתונים של הארגון, זאת באמצעות הכרה יסודית של המערכת

הקיימת ואומדן המידע שנדרש אך אינו מסופק על־ידי המערכת הקיימת. כן יש לאסוף מידע על עלות המערכת הקיימת לצורך בדיקת הכדאיות הכלכלית.
קבוצת העבודה "מרכיבה" מתוך הנתונים מספר מערכות ומפרטת לגביהן עלויות ומידת־מילוי דרישות הארגון. מתוך המערכות הללו בוחרים את הפתרון הטוב ביותר, שנבחן ביסודיות מבחינת כדאיות, התאמה, והשפעה על הארגון.
מעריכים את השפעתה של החלפת המערכת של עובדי הארגון. השפעה זו כוללת הדרכה, ייעוץ והתאמת כוח העבודה על העלויות המתאימות.
בחינת הכדאיות הכלכלית מסכמת את יתרונות המחשב, העלות הצפויה והחסכונות הצפויים באם קיימים.
טבלה 4.2, מראה ניתוח מתומצת. המספרים אינם מדויקים משום שהציוד הספציפי עדיין לא נבחר. לצורך בדיקת הכדאיות אומדים את אורך חיי המערכת ב־5—8 שנים. 6 שנים מהוות אומדן משביע רצון אלא אם כן מתעוררות בעיות המשנות את המצב.
תוצאות בדיקת הכדאיות בתוספת המלצות קבוצת העבודה מוגשות לאישור ההנהלה. השלב הבא לאחר האישור הינו הכנת מפרט.

הכנת מפרט וקבלת הצעות

המפרט הוא מסמך מפורט המתאר את דרישות הארגון מהמערכת המוצעת. רוב המידע שנאסף עבורו משמש למטרות בדיקת הכדאיות. כן משמש המפרט הן כסיכום פנימי של המערכת המוצעת והן לשם קבלת הצעות מספקי ציוד עיבוד נתונים. דוגמא בטבלה 4.3.
כל ספק מקבל העתק של המפרט ומגיש הצעה הכוללת בדרך כלל את הסעיפים המאוזכרים בטבלה 4.4.
החוזה עם הספק מבוסס בדרך כלל על חוזה סטנדרטי של הספק, אך עדיין נותרות נקודות רבות למיקוח. לדוגמא: הסיוע הטכני הניתן על־ידי הספק (כגון: מספר מנתחי המערכות שיפעילו את המערכת).

בחינת ההצעות

שיטה מקובלת לבחינת הצעות מוצגת בטבלה 4.5.
ההצעות מוערכות על בסיס ציונים לפי מספר קריטריונים וסיכומם לגבי כל הצעה. אפשר לתת משקל יחסי לקריטריונים השונים על־ידי חלוקת ציונים בהתאם.
קריטריונים שונים יקבלו משקל יחסי שונה מארגון לארגון. השאלות המתעוררות הן: האם הציוד יעמוד ויפעל באילוצי הזמן, מה תהיה עלותו היחסית ומה מידת גמישותו להרחבה בעתיד. מערכת צריכה לאפשר גידול — בגודל קבצים, במספר ובמורכבות יישומים, ובמנין הפעולות. ההערכה צריכה להתחשב לא רק בתוכנה הקיימת אלא גם באפשרויות לקבלת תוכנה בעתיד.
קיימת אפשרות לרכוש גם מערכת מעורבת. חברות אחדות מציעות ציוד היקפי

[illegible] שביצועיו טובים יותר ובמחיר נמוך יותר מאשר [illegible] של הארגון. [illegible] הוא הגורם המגבלי הנוצר כתוצאה [illegible]

[illegible] לבדוק את [illegible] הביצוע המצויינים בהצעת הספק ולבדוק [illegible] את ביצועי המחשב. הקריטריון הנכון לבחינת ביצועי [illegible] (throughput) אשר לא ניתנת לבדיקה מראש משום שלשם [illegible] את כל התהליכים ולהריצן.

[illegible] בדיקה מקובלות:

שיטת הבדיקה	הערות
1. מחזור זיכרון (core cycle time)	מדידה כלל לא מדויקת. משמעותי רק במערכות בעלות ארגון דומה של זיכרון.
2. זמן ביצוע הוראות (instruction time)	מדידה לא מדויקת משום שיש להתחשב בתדירות השימוש בכל הוראה.
3. זמן ממוצע לקבוצת הוראות מייצגות (instruction mix times)	זמן ממוצע לביצוע הוראה המבוסס על התדירות הצפויה לשימוש בהוראות השונות. מדד גס, אך טוב יותר מ־2.
4. Kernel Problem	בעיות מוצגות מקודדות בהוראות המערכת, מורצות, וזמני הביצוע נמדדים. יעיל לחישובים מתימאטיים סטנדרטיים. משקף את היכולת החישובית של המחשב אך אינו מביא בחשבון יחידות קלט פלט.
5. בעיות Benchmark	בעיות סטנדרטיות מהסוגים המבוצעים במציאות. מקודדים את התוכניות ומעריכים או מודדים את זמני הביצוע. בעיות ה־benchmark מייצגות נכונה את הבעיות במציאות, אך אינן מדגימות את כל שלבי העיבוד. זו השיטה המקובלת ביותר.
6. סימולציה (simulation)	תכונות המערכת המוצעת מושוות עם תכונות מערכות אחרות. ביצועי המערכות מוערכים על סמך סימולציה [illegible] במחשב.

הערכת יישומי מערכת המידע

ניתן לבדוק את יישומי מערכת המידע בשלושה מדדים: טכני, תפעולי, וכלכלי. בעבור יישומים חדשים מהווים המדדים הללו בדיקות כדאיות, בעבור היישומים הקיימים הם מהווים מדדי ביצוע.

1. בדיקת כדאיות / ביצוע טכני — האם ניתן לבצע את היישום בטכנולוגיה הקיימת?
2. בדיקת כדאיות / ביצוע תפעולי — האם המערכת פועלת (תפעל) בהצלחה? האם משתמשים (ישתמשו) בה?
3. בדיקת כדאיות / ביצוע כלכלי — באיזה מידה רבה התועלת מהעלות של המערכת?

בדיקת כדאיות טכנית

בדיקת כדאיות טכנית בוחנת אם השיטה המוצעת לעיבוד נתונים אפשרית. יישומים רבים אינם ניתנים לביצוע בציוד ובתוכנה העומדים לרשות הארגון. במסגרת בדיקה זאת נשאלות השאלות הבאות:

— האם מספיקה מהירות העברת המידע?
— האם יש זיכרון חיצוני מספיק לאחסון הנתונים?
— האם יחידת העיבוד המרכזית יכולה לענות על הדרישות בזמן נתון?
— האם קיימות שיטות פתרון לכל הבעיות?
— האם מערכת ההפעלה מותאמת לגישה התפעולית?

לאחר הפעלת יישומים מסוימים, ייתכן שהערכה מחדש תגלה חוסר־יעילות הנובע מאי־יכולת הציוד והתוכנה לסייע כנדרש ליישומים הללו. ייתכן ופעולה תעובד בזמן אמיתי (on-line — real-time) אך באיטיות רבה, משום שלמחשב הנדון אין יכולת לעמוד בעומס.

בדיקת כדאיות תפעולית

שיקולי בדיקת הכדאיות התפעולית מתייחסים לשאלות:

האם ניתן לספק את נתוני הקלט? האם ניתן להשתמש בפלט?

לדוגמא: קיימת אפשרות טכנית שסוכני המכירות ידווחו על כל מכירה באמצעות טלפון, אך מבחינה תפעולית אין זה מעשי. קיימת אפשרות טכנית להדפיס דוחות מחשב של 100 עמודים, אך אין אפשרות תפעולית יעילה לנצלם. ניתן למקם מסוף במשרדו של כל מנהל, אך הניסיון מראה כי השימוש בו מצומצם ביותר.

הערכה של יישומים לאחר הפעלתם בוחנת בעיקר את מידת יעילותם, אחוז שיגויים, נצולת וכו׳. מחקרים הראו כי חשוב להעריך בפרקי זמן קבועים את מידת היעילות של דוחות המחשב. קיימת נטיה להמשיך פרסום דוח קיים,

לראות שאינו נמצא בשימוש שוטף, משום שקיימת סברה כי יועיל בעתיד.

השיטות לגילוי תופעה זו הן:

— הפסקת פרסום הדוח כדי לראות אם יבקשוהו לכשלא יגיע.

— סקירה תקופתית של הדוחות המופצים.

— העמסת עלויות אשר תניע מנהלים לצמצם ככל האפשר בפרסום דוחות.

בדיקת כדאיות כלכלית

בעת בחינת הצעה יש לערוך בדיקת כדאיות. לאחר התקנת המערכת יש להעריך מדי פעם בפעם את היחס של העלות/תועלת (cost/benefit).

ניתן לחלק את העלויות והתועלויות של מערכת עיבוד נתונים כדלהלן:

1. עלויות:
   א. נמדדות או נאמדות עם שונות נמוכה.
   ב. נאמדות עם שונות בינונית או גבוהה.
2. תועלויות:
   א. נמדדות או נאמדות עם שונות נמוכה.
   ב. נאמדות עם שונות בינונית או גבוהה.

דוגמאות ניתנות בטבלה 4.6.

הוצאות ההפעלה השוטפת של מערכת ידועות כמעט בוודאות ולכן נכללות הן בקבוצת העלויות הנאמדות עם שונות נמוכה. אמידת העלויות שיש להעמיס על יישום בודד קשה ביותר משום שחלק מההוצאות משותפות ליישומים אחרים.

ניתן לאמוד את העלויות הקשורות לפיתוח יישומים, אך שונותן נעה בדרך כלל בין 25% ל־100%. ישנן מספר עלויות משמעותיות, שנוהגים לזונחן משום ששונות אומדניהן גבוהה מאוד והן משתייכות יותר למחלקות ארגוניות אחרות מאשר לעיבוד נתונים. לדוגמא:

עלויות הרצת המערכת — העלויות הקשורות בהכנסת המערכת לשימוש הן משמעותיות לעתים, אך אינן מדווחות: הדרכת משתמשים בזמן עבודתם הרגילה, וכן ירידת יעילותם במשך תקופת ההדרכה.

התנגדות עובדים — התנגדותם או אי שביעות רצונם של העובדים למערכת החדשה יכולה לגרום לירידת יעילות עבודתם.

טיפול בשיגויים — עלות הטיפול בשיגויים — בדיקה, תיקון, והרצה מחדש — משמעותית אך קשה לאמידה משום שהיא מפוזרת בארגון.

אובדן הכנסות כתוצאה משגיאות בעיבוד — אי שביעות רצונם של הלקוחות הנובעת מרמת שגיאות גבוהה קשה לאמידה, אך קיימת.

טבלה 4.2

ניתוח העלות

אומדן העלות של מערכת מהשב חדשה		
עלות הכנת המבנה הפיזי	× ×	
בחינה ותכנות של היישומים הבסיסיים	× ×	
עלות הדרכת כוח־אדם	× ×	
התאמת קבצים והפעלה במקביל	× ×	
סה״כ הוצאות חד־פעמיות		× ×
אומדן עלות הפעלה שנתית		
הוצאות שכירה או הכירה, ותחזוקה	× ×	
שכירת תוכנה	× ×	
מתכנתים ומתכננים	× ×	
מפעילים	× ×	
חשמל, שכר דירה וכו׳	× ×	
סה״כ הוצאות הפעלה		× ×
חיסכון שנתי (הוצאות נחסכות פחות הוצאות הפעלה שנתיות)		× ×
שיעור תשואה (השער שבו הערך הנוכחי הנקי של החסכונות השנתיים שווה להוצאה החד־פעמית)		× ×%
יתרונות לא כספיים אחרים.		

טבלה 4.3

דוגמא למפרט של הארגון

1. מבוא:
   א. תיאור הארגון ופעולתו.
   ב. סכום הדרישות.
   ג. ציוד נוכחי.
   ד. תהליך הבחירה — הקריטריונים לפיהם ייבחן הציוד.
2. דרישות מהמערכת:
   א. תכונות ציוד דרושות (hardware).
   ב. תכונות תוכנה דרושות:
      1. קומפילרים (פורטרן, קובול, R.P.G.).
      2. חבילות תוכנה (utility packages).

3. חבילות יישומיות (application packages, כמו תכנות לינארי).
4. מערכת הפעלה.
5. מערכת ניהול נתונים (data management system).

ג. סיוע נדרש:
1. אנשי מערכת הדרושים מהספק.
2. מערכת לגיבוי (backup facilities).
3. אמצעים הדרושים לבחינה.

ד. אילוצים:
1. מועד מתוכנן לאספקת הציוד והתוכנה.
2. אילוצי זמן בביצוע עיבוד הנתונים.

ה. אפשרות צמיחה בעתיד.

3. יישומים עיקריים (לכל יישום צריכים להתקיים הסעיפים הבאים):
א. תיאור המערכת.
ב. תיאור קבצים; כולל גודל נוכחי וקצב גידול.
ג. פירוט מבנה הקלט וכמותו.
ד. תרשימי זרימה של המערכת וכן תיאור של:
— תדירות העיבוד.
— מספר הפעולות.
— שיטת העיבוד המוצעת.

טבלה 4.4

סעיפים בהצעת ספק של מערכת

1. הרכב הציוד (equipment configuration):
א. יחידות הציוד.
ב. נתוני הפעלה וביצוע של הציוד.
ג. הרכבים אלטרנטיביים.
ד. גמישות המערכת — יכולת הרחבה בעתיד (מודולריות).
ה. דרישות מיוחדות לגבי מקום ועלויות התקנה שונות.

2. עלות המערכת המוצעת:
א. עלות רכישה ו/או שכירה.
ב. עלות נוספת עבור משמרות נוספות.
ג. חוזה תחזוקת הציוד.
ד. שכירה או רכישת תוכנה.

3. תוכנה קיימת, וחבילות תוכנה.

4. סיוע למערכת:
א. ניתוח מערכות הכלול במחיר: עלות עבור שרותים נוספים.
ב. שרותי תכנות הכלולים במחיר: עלות עבור שרותים נוספים.
ג. סיוע ההדרכה ועלויות בהתאם.
ד. תנאי שימוש במערכת אלטרנטיבית בעת־הצורך.

5. תנאים:
א. תאריך אפשרי למסירת המערכת.
ב. תנאי תשלום.
ג. אופציות שכירה/קנייה.
ד. זמן ניסוי הכלול במחיר.

6. ביצועי המערכת עבור היישומים במפרט:
א. שינויים מהמערכת המוצגת במפרט.
ב. זמני ביצוע.
ג. שינויים בזמני ביצוע כתוצאה מהוספת פריטי ציוד שונים.

7. אינפורמציה נוספת.

טבלה 4.5

קריטריון להערכה	נקודות אפשריות	נקודות שקיבלו הצעות שונות א.	ב.	ג.
1. ביצוע המערכת (42%):				
א. ביצועי ציוד	60	60	41	42
ב. ביצועי תוכנה	60	39	43	31
ג. אפשרויות הגדלה והרחבה	55	41	48	30
2. תנאי ספק (25%):				
א. מוניטין של ספק	40	25	32	21
ב. תחזוקה	20	14	18	13
ג. סיוע למערכת	20	12	18	12
ד. העדפות עובדים	25	15	20	5
3. עלות (33%):				
א. עלות שכירה	50	35	23	40
ב. תנאים לשימוש ממושך	20	17	5	18
ג. עלות הדרכת עובדים	25	21	9	17
ד. עלות תחזוקה	20	15	10	14
ה. הוצאות חד־פעמיות	20	18	11	14
	415	312	278	257

טבלה 4.6

	עלויות	תועלות
שונות נמוכה	ציוד תוכנה נקנית התקנת המערכת כוח־אדם — מפעילים אספקה — ניר וכו׳ תחזוקה	חסכונות הנובעים מצמצומים בכח אדם.
שונות בינונית	פיתוח יישומים	חסכונות הנובעים מצמצומים בנכסים כגון מלאי. שיפורים בניהול נכסים, גידול בתשואות, צמצום בהלוואות.
שונות גבוהה	"הרצת" המערכת הקטנת יעילות כתוצאה מהתנגדות עובדים. טיפול בשיגויים אובדן פדיון כתוצאה משגיאות	גידול בפדיון כתוצאה מצמצום זמן שרות והרחבת אפשרויות גישה למידע. צמצום עלויות כתוצאה משיפור דיוק. ביצועים טובים יותר הנובעים ממידע מעודכן יותר והרחבת האמצעים לקבלת החלטות.

3.

תהליך בחירת מחשב ושיטות מימון

מבוא

תהליכי בדיקת הכדאיות וההחלטה לרכוש מחשב כרוכים ב־8 צעדים בסיסיים:

1. הכנת תוכנית בדיקה הכוללת "הצדקה" ראשונית לשימוש במחשב. בהצדקה זו יש לפרט את הדרישות לעיבוד נתונים ולקבוע יתרונות ערכיים וכמותיים שיצמחו משימוש במערכת המחשב באופן כללי.
2. ניתוח מערכת עיבוד הנתונים הקיימת.
3. קביעת שיטות מימון הרכישה.
4. ניתוח מפורט של המערכת המוצעת (כולל כל שלבי הבחירה של המערכת הספציפית).
5. החלטה על הסוג הספציפי של הציוד.
6. פירוט מערכת העלויות השוטפות והקבועות, הוצאות אחזקה וכו׳.
7. הערכה של תוצאות הבדיקה.
8. הכנת המלצות סופיות להנהלה.

נתרכז בשלבים 3—5, הדנים בניתוח והכוונת תהליך בחירת המחשב ושיטות המימון השונות. ננסה לקבוע מהלך מחשבה עקבי שיכלול שלבי החלטות וזרימת אינפורמציה בכל הקשור לתהליך בחירת המחשב. ישנה חשיבות בהצגת תהליך זרימת האינפורמציה וההחלטות מהשלב שבו הוגדרה כללית ההצדקה הראשונית לשימוש במערכת עיבוד נתונים אוטומטית עד להחלטה על שיטת מימון הרכישה של ציוד ספציפי.

השלב הראשון בתהליך הניתוח הוא, כאמור, הגדרת הבעיה, או במלים אחרות, הצדקת המעבר למערכת עיבוד נתונים אוטומטית. איש עיבוד הנתונים בארגון חייב בשלב זה להצדיק את השימוש במערכת עיבוד נתונים אוטומטית, כדי

שיוכל לעבור לשלב השני הכרוך בבחירה. הבעיה הראשונית אינה פשוטה כלל ועיקר ועל איש עיבוד נתונים אוטומטית להתחשב במכלול הגורמים הקיימים ביחידה שבה הוא מועסק ולקבוע דרישות לעיבוד נתונים אוטומטית בטווח הקצר ובטווח הארוך.

בשלב זה כבר נקבע אלמנט ראשון של הבחירה בכך, שמחליטים מראש על סוג הבעיות הקיימות, אשר מכתיבות את הצרכים לעיבוד נתונים אוטומטית. באופן כללי ניתן לחלק את השימושים במחשב לשלוש קבוצות ראשיות:

— שימושים טכניים.
— שימושים אדמיניסטרטיביים.
— שימושים מדעיים.

ייתכן מאד שבמוסד מסוים קיימים שימושים אחדים, כגון באוניברסיטה, בה רבים השימושים המדעיים והשימושים האדמיניסטרטיביים, דבר שיכתיב במידה רבה את תהליך הבחירה.

לאחר שנקבעה מערכת השימושים תוך הדגשת היתרונות והחסרונות לטווח ארוך ולטווח הקצר, יש מקום להציג בשלב הבא ניתוח של מערכת עיבוד הנתונים הקיימת (לדוגמא: הנהלת חשבונות ידנית או במכונות חישוב מיכאניות ואלק־טרוניות), תוך הצגת מגבלותיה, יתרונותיה, וכו׳.

איננו עוסקים בניתוח כל המערכת ועל כן לא נתעכב על השלבים 1, 2 של התהליך. ברם, יש להדגיש את החשיבות המירבית של הדיון וההצגה של שלבים אלו, היות והשוואה הערכית והכמותית של דרישות ממערכת חדשה בהשוואה למערכת הקיימת יכולה לקצר את כל תהליך הניתוח. בהרבה מקרים עשוי הניתוח להסתיים כבר בשלב זה, כיוון שמתברר שאין הארגון זקוק למערכת עיבוד נתונים אוטומטית, או שהמערכת הקיימת עונה על הדרישות לתקופה נוספת.

תהליך הבחירה של המחשב (תרשים 4.6)

מערכת השיקולים בבחירת מחשב מורכבת מאד ושיקולי הבחירה נעשים בין מכלול גורמים אובייקטיביים וסובייקטיביים מהד, וגורמים ערכיים וכמותיים מאידך. מערכת שיקולים אשר אינה מלווה בעקביות מחשבתית וביצועית כאחת לוקה בחסר בכך, שאינה מדריכה את האדם האחראי להצגת שיקולי הבחירה ומקשה על מקבל ההחלטות.

אין הכוונה כאן להציג כלי החלטה יישומי לבחירת מחשב, אלא לנסות לבנות מודל של זרימת אינפורמציה והחלטות, העשוי להקל על איש עיבוד הנתונים בעריכת מכלול השיקולים בצורה מסודרת ועקבית, כך שלבסוף יוכל להביא את כל הגורמים אל "נקודה סופית" שבכוחה לקבוע את ההחלטה.

בגמר השלב בו הוצגה ההצדקה הראשונית לבחירת המחשב ונקבעו קבוצות השימוש העיקריות, על המנתח לקבל 4 סוגי החלטות עיקריות:

א. החלטה על סוג הבחירה.
ב. החלטה על קבלת ייעוץ.
ג. החלטה על שיטת הבחירה.
ד. החלטה על סל הקריטריונים הרלוואנטיים ומשקליהם.

ארבע החלטות עיקריות אלו אינן תלויות זו בזו והן יכולות להתקבל במקביל. ההחלטות קובעות כמובן "מערך" של עובדות, נתונים ושאלות המתחברים למערך החלטה הקובע את תהליך הבחירה. בכך נקבעים חקר השווקים, סוגי המבחנים, השאלות הספציפיות וכד', אשר אמורים לתת לבסוף ערך כמותי ו/או ערכי לכל מערכת מחשב העומדת לבחירה בצירוף אלטרנטיבות אחרות.
הסדר הכרונולוגי של ההחלטות אינו הכרחי והחלטה אחת אינה תלויה באחרת, אם כי ההיגיון מחייב, שההחלטה על סוג הבחירה וסוג הייעוץ ייקבעו תחילה, ולאחר מכן יוחלט על שיטת הבחירה וסל הקריטריונים.

האלטרנטיבות המשתתפות בכל קבוצת החלטה מובאות להלן:

א. סוגי הבחירה:
1. קביעה מראש של מקור (יצרן) אחד רצוי ביותר.
2. מכרז תחרותי בין מספר יצרנים.

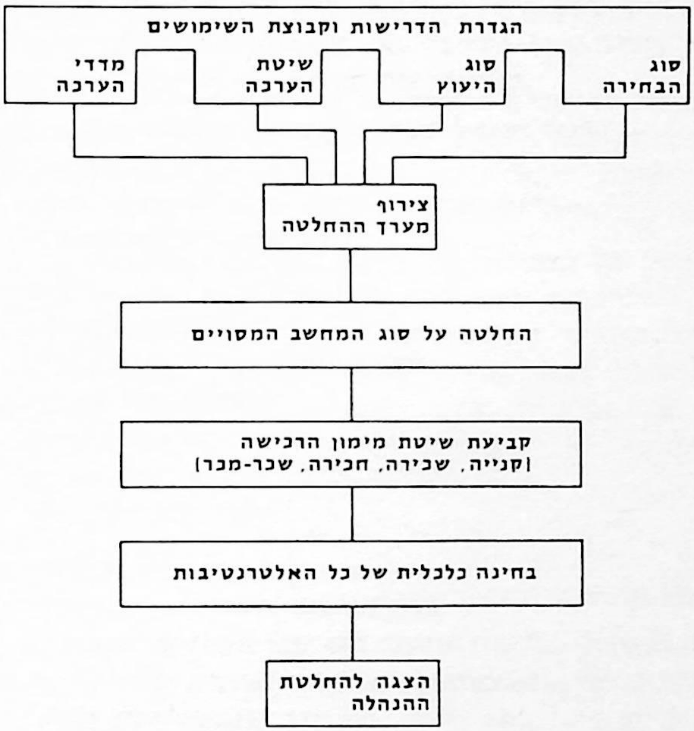
ב. סוגי הייעוץ:
1. בלי כל עזרה מבחוץ.
2. יועצים עצמאיים — מומחים בנושא.
3. פירמות ייעוץ מתמחות.
4. פירמות של יועצי ניהול.
5. פירמות ייעוץ טכניות (הנדסת מערכות, וכד').
6. ייעוץ של יצרני הציוד.

ג. שיטות הערכה:
1. הערכה של Benchmark Problems.
2. הערכה על פי דוחות לגבי טיב התוכנה והציוד.
3. הערכה על פי תוכניות ניסוי (test problems).
4. שיטות סימולציה.
5. מודלים מתימאטיים.

ד. מדדי הערכה:
1. קיבולת וביצוע הציוד.
2. ביצועי התוכנה.
3. הקשר עם היצרן (שירות, עזרה, הדרכה, וכו').
4. אפשרויות היישום של התכניות.

5. מידת ההתאמה הפנימית של הציוד והתוכנה.
6. זמן האספקה.
7. מודולריות של הציוד ופוטנציאל ההרחבה.
8. עלויות — בחינה כלכלית.

הערות:
1. בהחלטה על סוגי ייעוץ ושיטות בחירה, האלטרנטיבות אינן בלתי תלויות וייתכן מאד שההחלטה תקבע שתי אלטרנטיבות או יותר למערכת השיקולים.
2. גודל "סל" הקריטריונים ייקבע בהתאם לספציפיקציות של הצרכן, ומשקלו של כל קריטריון ייקבע סובייקטיבית בהתאם לעוצמת חשיבותו בסה"כ שיקולי הצרכן.



תרשים 4.6: תהליך בחירת מחשב

שיטת מימון המחשב

תעשיית המחשבים החלה בהשכרת ציוד. רק לאחר שנת 1965 התחילה חברת י.ב.מ. למכור ציוד מחשבים. עד היום מעודדות חברות המחשבים השכרת ציוד, דבר המעודד מכירות, שכן הלקוח אינו צריך להשקיע בתחילה השקעה גדולה.

למעלה משני שלישים מהמחשבים המצויים כיום בעולם שכורים; בישראל כמעט כל המחשבים בסדרי גודל בינוניים וגדולים הם שכורים. כמה סיבות לכך והעיקרית שבהן בעית ההתיישנות הטכנולוגית של הציוד. על־ידי השכרת המחשב ניתן להחליפו בכל זמן, דבר הקשה יותר כאשר המחשב קנוי. אולם, כפי שיתואר להלן, נראה בכל זאת כי האופציה הטובה ביותר עשויה להיות רכישת המחשב ולא שכירתו, או אולי שכירתו מיד שלישית, כמקובל בארה״ב ובאירופה *.

הבסיס להחלטה צריך להיות כלכלי ומבוסס על הערך הנוכחי המינימלי של תזרים המזומנים הרלוואנטי לאחר מס.

קיימים כמובן גורמים אחרים אשר צריכים להילקח בחשבון:

א. **מיקוח**: אם הציוד הוא שכור קיימת האפשרות להפעלת לחץ על היצרן לספק שירותים טובים יותר; במקרה שהציוד קנוי אין קיימת אפשרות זאת. חברות המחשבים מעונינות בהשכרת ציוד יותר מאשר במכירתו, לפיכך מושפע בדרך כלל השירות הניתן מצורת ההתקשרות עם היצרן.

ב. **התיישנות**: הטיעון העיקרי בעת שכירת מחשב הוא התיישנותם של מחשבים. כאשר שוכרים ציוד ניתן בקלות להחליף את המחשב בהופיע ציוד חדש. בישראל שונה הדבר במידה מסוימת עקב שיטת תשלומי המכס המאפשרת החלפת ציוד רק בזמנים קבועים לפי תשלומים הנקבעים על־ידי המכס. אולם יש להביא בחשבון כי החלפה תדירה אינה רצויה אם ההשקעה בתכנון המערכת (כולל ניתוח, תיכון ותכנות) היתה גדולה וטרם כוסתה.

שלושה חלקים להתיישנות:

1. **התיישנות פיזית**: כאשר המחשב מתיישן מבחינה פיזית יש להחליפו. אולם רוב המחשבים בנויים כך, שאורך החיים שלהם מספיק לכיסוי מחיר הקנייה. אורך החיים הפיזי של מערכות מחשבים תלוי במרכיבים והוא נע בין 8 ל־14 שנים. התיישנות פיזית אינה צריכה, אפוא, להילקח בחשבון אם נקודת האיזון בין האופציות השונות היא פחות מ־8 שנים.

2. **התיישנות טכנולוגית**: תעשיית המחשבים עברה זעזועים טכנולוגיים לאורך ההיסטוריה הקצרה יחסית שלה — כ־25 שנה. שינויים אלה מדור לדור של

---

* שכירות יד שלישית התפתחה בשנים האחרונות. גוף מסוים (יד שלישית) רוכש את המחשב מיצרן המחשבים ומשכירו במחירים זולים יותר מאשר מחירי היצרן. שיטה זאת מבוססת על העיקרון שמשך החיים של המחשב ארוך יותר ממה שקובעות חברות המחשבים.

מחשבים שיפרו את יעילותם ואת ביצועיהם. אולם, הואיל והיצרנים השקיעו כספים ומאמצים בפיתוח מערכות חדשות, יש להניח שעד שלא יכוסו כל ההש־קעות לא יהיה כדאי להם לצאת עם מערכות חדשות. לפיכך, מרחק הזמן בין דור לדור הוא כ־6 שנים לפחות. תופעה זאת אופיינית במיוחד לשוק המחשבים. נוסף לכך, קצב השינויים הטכנולוגיים קטן עם הזמן. נראה שקיים גבול לשיפורים שאפשר להשיג. לכן עניין ההתיישנות הטכנולוגית איבד מערכו בשנים האחרונות, ובמקרה של קנייה הסיכון הוא קטן יותר ממה שנראה על פני השטח.
חשוב ביותר להדגיש כי שינויים טכנולוגיים אינם צריכים בהכרח להשפיע על הלקוח כל עוד המערכת שברשותו עדיין עונה לדרישות והמצב האובייקטיבי אינו מחייב התרחבות או שינוי.

3. **התיישנות כלכלית**: הגורם האחרון שצריך להילקח בחשבון הוא ההתיישנות הכלכלית — לאמור, נקודת הזמן שבה לא יכולה עוד מערכת המחשב לענות לדרישות הלקוח בצורה יעילה. נקודת זמן זאת נקבעת על־ידי ההשקעה בתכנון יותר מאשר על־ידי מחיר הציוד. בדרך כלל, מערכת צריכה להיות מתוכננת בצורה אשר תאפשר כיסוי כל הוצאות התכנון (ניתוח ותיכון, תכנות, בדיקות וכד׳). ברוב הארגונים והחברות עלות הפיתוח והתכנון של מערכות עולה על מחיר הציוד, לפיכך עובר אורך החיים הכלכלי של המערכות את נקודת האיזון שבין שכירות לקנייה.

גורמים כלכליים

המרכיבים העיקריים של העלויות באופציות השונות הם כדלהלן:

1. שכירות:
   - תשלומי שכירות חודשיים.
   - תשלומים נוספים עבור שימוש מעל משמרת אחת (בדרך כלל משמרת אחת = 176 שעות לחודש).
   - תשלומים עבור מכס, מס קנייה ומס שירותים.

2. שכירות יד שלישית:
   - תשלומי שכירות חודשיים (בלי שעות נוספות).
   - אחזקה (בדרך כלל בתוך תשלומי השכירות).
   - מכס, מס קנייה ומס שירותים.

3. קנייה:
   - השקעה ראשונית בקניית הציוד.
   - אחזקה (חוץ מתקופת האחריות, שהיא בדרך כלל כשלושה עד ששה חודשים).

— ביטוח.
— אפשרות שימוש בפחת מואץ לצרכי מס הכנסה.

4. קנייה בתשלומים:
— קיימת אפשרות לקנות את הציוד בתשלומים.
— יתר המרכיבים כמו בקנייה.

יתר המרכיבים הכלכליים, כגון עלות הבניין, מיסוי רגיל וכד׳, שווים בכל האופציות.

לשם ניתוח והשוואת האלטרנטיבות השונות, נשתמש בשיטת הערך הנוכחי של זרם המזומנים. בשיטה זאת ניתן להשוות את האלטרנטיבות השונות ולהגדיר את נקודת הזמן אשר לאחריה רכישת מחשב כדאית יותר, כלומר, אם זמן השימוש עובר נקודה זאת נרכוש את המחשב. כדי לבצע את הניתוח נבדקו כ־30 הצעות של חברות מחשבים שונים, שהביאו למסקנות הבאות:

מחיר רכישת מערכת מחשב* שווה ל־48 תשלומי שכירות חודשיים. אחזקה חודשית במקרה של קנייה שווה ל־10% דמי שכירות חודשיים. פרמיית ביטוח במקרה של קנייה שווה ל־1% דמי שכירות חודשיים. שכירות חודשית מיד שלישית שווה ל־85% דמי שכירות חודשיים רגילים.

הניתוח

כדי להשוות את האלטרנטיבות השונות שהוצגו לעיל השתמשנו בשיטת הערך הנוכחי; לביצוע ההשוואה נבחר מחיר הון של 8%.

כדי שהניתוח דלהלן יהיה כללי כל האפשר, נבחרו דמי שכירות חדשיים של 1 $ כבסים לכל האלטרנטיבות השונות, כך שמחיר הרכישה הוא 48 $, אחזקה 0.1 $ ביטוח 0.01 $ ושכירות יד שלישית 0.85 $.

במקרה הכללי, כאשר אין מכס ומסים אחרים (כמקובל בארה״ב), מתקבלות התוצאות כפי שמוצגות בטבלה 4.7. טבלה זאת מציגה את הערך הנוכחי (לפי 8%) בדולרים לשלושת האופציות של שכירות, קנייה ושכירות יד שלישית.

---

* מחירי רכישה של מרכיבים שונים במערכת המחשב נעים בין 30 ל־60 חודשי שכירות. אולם, כאשר מנתחים מערכות כוללות (מערכת מחשב מלאה), היחס הוא בערך 48 חודשי שכירות. הנתונים האלו, הם, כמובן, ממוצעים, והם מבוססים על ניתוח מספר גדול של הצעות.

טבלה 4.7

ערך נוכחי — השוואה בדולרים אמריקאיים *

שנה	שכירות	קנייה	קנייה+ערך גרט	שכירות יד שלישית
1	11.57	49.27	13.58	9.83
2	22.25	50.45	20.50	18.91
3	32.12	51.53	25.85	27.30
4	41.23	52.53	32.52	35.05
5	49.64	53.46	36.61	42.20
6	57.41	54.31	43.43	48.80
8	64.58	55.10	47.94	54.90
8	71.20	55.83	50.53	60.52
9	77.31	56.50	54.00	65.72
10	82.95	57.12	54.86	70.52

מטבלה 4.7 מתקבלות המסקנות הבאות:

א. קנייה בלי ערך גרט עדיפה על שכירות לאחר 6—5 שנים.

ב. קנייה עם ערך גרט עדיפה על שכירות לאחר 2—1 שנים.

ג. שכירות יד שלישית עדיפה על שכירות רגילה.

ד. קנייה עם ערך גרט עדיפה על שכירות יד שלישית לאחר 3—2 שנים.

טבלה 4.8 מציגה את שלוש האלטרנטיבות לאחר ניכוי מס. ההנחה היא כי לארגון מפעיל המחשב יש הכנסות ולכן יש משמעות לניכוי מס (ניכוי מס שווה להוצאה מוכרת למס כפול שיעור המס). ככל ששיעור המס גבוה יותר, הניכוי גבוה יותר. כתוצאה מכך העלות לאחר מס נמוכה יותר ככל ששיעור המס גבוה יותר.

ההוצאות המוכרות למס כוללות: פחת, הוצאות מימון, עלות־ביטוח, מס רכוש וכד׳. בטבלה מס. 4.8 לא נעשית השוואה למקרה של קנייה עם ערך גרט, אולם יש לבדוק אלטרנטיבה זאת בכל מקרה ספציפי.

* שכירות = 1 $, קניה = 48 $, שכירות יד שלישית = 0.85 $, אחזקה במקרה של קנייה = 0.1 $, ביטוח במקרה של קנייה 0.01 $ לחודש. ערך הגרט נלקח בחשבון בניתוח ל־10 השנים באחוזי מערך הקנייה: 75, 70, 65, 55, 50, 35, 25, 20, 10, 0, בהתאמה.

טבלה 4.8

ערך נוכחי — השוואה בדולרים אמריקאיים *

עם ניכוי מס

שנה	שכירות	קנייה	שכירות יד שלישית
1	5.38	44.87	4.57
2	10.35	41.99	8.79
3	14.94	39.33	12.69
4	19.17	36.87	16.29
5	23.08	34.60	19.62
6	26.69	32.50	22.69
7	30.03	31.33	25.52
8	33.11	31.67	28.11
9	35.95	31.99	30.56
10	38.58	32.27	32.79

מטבלה מס. 4.8 מתקבלות המסקנות הבאות:

א. קנייה עדיפה על שכירות רגילה לאחר 8—7 שנים.

ב. שכירות יד שלישית עדיפה על שכירות רגילה.

אם נשווה את התוצאות בטבלה מס. 4.8 לאלו שבטבלה מס. 4.7, נראה שניכוי המס משפר למעשה את האלטרנטיבה של שכירות לגבי קנייה. תופעה זאת נובעת כמובן משיטת הפחת של 15% לשנה (קו־ישר) המותרת בחוק. בארצות אחרות מאפשרים השלטונות להשתמש בפחת מואץ המשפר בהרבה את אלטרנ־טיבת הקנייה. אילו השתמשנו בפחת מואץ, היינו מעדיפים את הקנייה על שכירות לאחר 6—5 שנים.

הצגנו כאן את הניתוח בצורה כללית ובדולרים. נמנענו מניתוח בלירות ישראליות עקב השינוי התכוף בערך הלירה הישראלית ובגלל השינויים הרבים הנעשים בתעריפי המכס. הניתוח המעשי צריך לקחת בחשבון את המרכיבים הבאים נוסף על אל שהוצגו לעיל:

— שער החליפין.

— תעריפי המכס.

* לצורך חישובי ניכוי מס נלקח שיעור המס החל על מפעל רגיל שהוא 53.5%. סה"כ זה כולל: מס חברות 35%, היטל בטחון 3% ומס הכנסה (25% מ־62% הנותרים) 15.5%. הפחת המוכר הוא 15% לשנה במקרה של קנייה.

— מס שרותים (בעיקר על אחזקה).
— ביטוח.
— דמי הובלה.
— שונות.

התוצאה המתקבלת מהחישובים דלעיל הנה עקבית ונובעת מהמסקנה האינ־טואיטיבית, שכאשר כל הגורמים שווים וככל שאורך החיים הכלכלי (או הפיזי) של ציוד המחשבים ארוך יותר, כן כדאית יותר האלטרנטיבה של רכישת הציוד. אולם, ככל ששער הריבית גבוה יותר, כן כדאית יותר אלטרנטיבת שכירת ציוד המחשבים.
מהנתונים השונים שהוצגו מתקבלת המסקנה, כי רכישת ציוד מחשבים, שעד לפני זמן מועט היתה בלתי מקובלת, נראית כיום מציאותית יותר. אולם יש להדגיש שבכל מקרה חייב להיעשות ניתוח כדאיות לגופו של עניין. בדרך כלל, אם המחשב המוצע עומד בדרישות הארגון תקופה של חמש שנים לפחות, כדאי יותר מבחינה כלכלית לרכוש את הציוד (בהנחה של ערך גרט).
יש לחזור ולהדגיש שניתוח זה העמיד אלטרנטיבות זו מול זו, בעוד זרם התש־לומים משמש כקריטריון להשוואה, אולם, במקרים רבים עדיין קיימים גורמים אשר אינם ניתנים לכימות, אולם צריכים להילקח בחשבון, כדוגמת טיב שירותי האחזקה, שינויי טכנולוגיים ועוד.

4.
היבטים שונים של שיתוף זמנים

מהם הקריטריונים להשוואת עיבודים בשיתוף זמנים (time-sharing) לעומת עיבוד בסדרות (batch-processing)? בהרבה מקרים, המטרות והדרישות של המערכת מחייבות שימוש במערכת של שיתוף זמנים. דוגמאות לכך — מערכת להזמנת מקומות לנוסעים בחברות תעופה, מערכות של סוכני בורסה למידע על שערי ניירות-ערך וכו׳. למערכות אלו משותפת הדרישה כי יש לאפשר למשתמש גישה מידית לנתונים; לתשובות לא תהיה משמעות אם המשתמש ייאלץ לחכות את הזמן הדרוש בתנאי עיבוד בסדרות. לעומת זאת, יש מקרים רבים שבהם קיימת אפשרות הבחירה בין שתי צורות העיבוד. אם זמן ההמתנה הנו גדול די הצורך לאפשר עיבוד בסדרות, אזי כל שיטה יכולה לספק תשובות ישימות.

נשווה עתה מערכת שיתוף זמנים למערכת קונבנציונלית של עיבוד בסדרות. במערכת שיתוף-זמנים שנדבר עליה להלן יש למשתמש גישה למחשב באמצעות מסוף, כלומר, הגישה למחשב מבוססת על הביקוש של המשתמש. זו המערכת בה קיימת אינטראקציה של אדם/מכונה הכרוכה בשני גורמים כלכליים יסודיים:

1. צרכנים רבים משתמשים באותו ציוד בו-זמנית.
2. הגורם העיקרי בפונקצית התמורה (payoff) של המשתמש הוא התועלת הכלכלית של זמן התגובה.

הביקוש של המשתמשים הוא בעיקרו אקראי ונתון לתנודות. המערכת מתאימה עצמה לתנודות, אולם לעתים עלולה להיות בטלה לתקופות קצרות, דבר המחייב את בעל המערכת לספוג הוצאות של זמן בטלה.

יש וזמן תגובה קצר מביא לתמורה גדולה ביותר. למשל, אם סוכן הבורסה יכול לספק תשובה מהירה לגבי שערי ניירות-ערך, סיכויו לבצע עסקה עבור הלקוח

גדלים. לעתים אין יתרון בתגובה מהירה וניתן לעבד בסדרות את השאילתות. למשל, אם הלקוח מבקש מסוכן הבורסה דוח על חברה מסוימת, ניתן לשלוח את הדוח יחד עם משלוח הדוחות לכל הלקוחות.
הבחירה בין עיבוד בשיתוף־זמנים לעיבוד בסדרות תלויה בסופו של דבר בשקילת גורמים הקשורים בצרכי המשתמש.

הגורמים בעד השימוש בשיתוף־זמנים הם:

1. יעילות משתמש רבה יותר.
2. יעילות של מחשב גדול.
3. גמישות ביישומים.

הגורמים בעד עיבוד בסדרות:

1. עלות המחשב.
2. עלויות תוכנה.
3. עלות המסוף.
4. עלויות התקשורת.

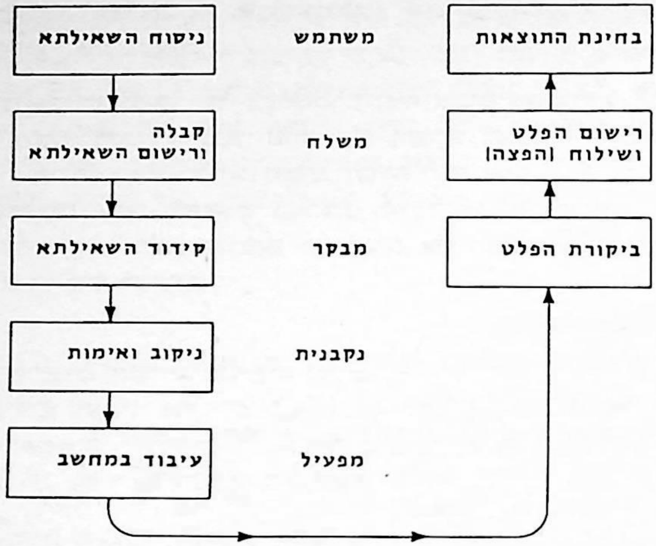
המשקל שהמשתמש נותן לכל אחד מהגורמים יקבע איזו שיטה עדיפה. בחירה בשיתוף זמנים פרושה שגורמי יעילות וגמישות של המשתמש חשובים ביותר מהעלות הגבוהה יותר עבור תוכנה, עבור ציוד נוסף ועבור מערכת תקשורת.

יעילות המשתמש — מושג זה משמש תכופות כטיעון בעד קישור המשתמש ישירות למחשב. קיימים מחקרים שמדווחים למשל, על שיפור של בין 3:1 ל־7:1 ביעילות המתכנת, בכתיבת וניפוי תוכניות, בעוד הוא נהנה מעבודה בשיתוף־זמנים.

ניתן להאמין לממצאים אלה אם ניקח באנלוגיה את היעילות הנמוכה של מנהל אילו השתמש בטלפון רק בשעות מוגדרות ביום במקום גישה ישירה למכשיר.

ידוע שהעלות לפעולת חישוב על מחשב גדול קטנה מזו של מחשב קטן ומכאן יעילותו של מחשב גדול. העובדה שהרבה צרכנים משתמשים בו־זמנית במחשב מחייבת שימוש במחשבים גדולים מרכזיים, דבר שמביא ליעילות חוזרת רבה עוד יותר. למחשב הגדול יש בדרך כלל גם מערכת תוכנה משוכללת יותר.

למערכת שיתוף־זמנים יתרון נוסף — היכולת להיראות כמכשיר קטן אינדי־בידואלי להרבה משתמשים, או כמחשב גדול מיוחד לשימוש הצרכן כאשר דבר זה דרוש לו. צרכן אחד יכול להשתמש בו כמכונת חישוב, בעוד שהשני ישתמש בו לצורך הרצת סימולציה מסובכת. על־ידי כך מושג היתרון החשוב של גמישות ביישומים עבור הצרכן, יתרון שאינו נמצא במערכת עיבוד בסדרות.
אחד מהיתרונות שמתעלמים ממנו לעתים בזמן השוואת שיתוף זמנים לעיבוד בסדרות כרוך בתהליך הכנת הקלט למחשב. ישנם יתרונות ברורים בהזנת הנתונים ישירות במקום הכנתם שלא תחת פיקוח יחידת העיבוד המרכזית (off-line).



תרשים 4.7: התהליך הקונוונציונאלי במילוי שאילתא

תרשים מס. 4.7 מראה את הצורה המסורתית של הזנת נתונים למחשב וקבלת תוצאות. בשיטה זו מעורבים הרבה אנשים ותהליכים רבים כרוכים בה. יש למלא טפסים, לנקב ולאמת כרטיסים, להעביר את המידע ולרושמו.

במערכת שיתוף־זמנים מבצע המחשב כמעט את כל המשימות הללו. המשתמש עצמו יכול להיות זה שליד המסוף והוא יהיה אז האדם היחיד בשרשרת. הקלט יכול להיות בשפת המשתמש והמחשב יכול לספק את הפלט ישירות למשתמש. אין צורך בניקוב (כמובן כשמדובר על כמויות גדולות של נתונים יהיה צורך בהכנתם, כולל ניקוב, בשיטת off־line). האימות נעשה על־ידי כך שהצרכן יכול לבחון את הקלט טרם שישחררו למחשב. המחשב יכול לבצע את כל פעולות הרישום. מערכת התקשורת והמסוף מטפלת בהעברת האינפורמציה. מערכת כזו תביא גם לפחות שיגויי קלט מאשר מערכת קונוונציונאלית מכיוון שיש בה פחות צעדים.

השוואת ערכים כמותיים כמו עלות ויעילות של מערכות עיבוד בסדרות ומערכות שיתוף־זמנים בתנאים שווים הנה בעיה קשה ביותר. אי־אפשר לבצע ניסויים מבוקרים, להתקין מערכת שיתוף־זמנים עבור קבוצת דרישות ואחר־כך להתקין מערכת עיבוד בסדרות עבור אותה קבוצת דרישות. לא תמיד ניתן לבצע כל אותן דרישות בשתי השיטות. שני גורמים הנם בעלי משמעות בקביעת יעילות מערכת שיתוף־זמנים:

1. האם המערכת מבצעת עיבוד בסדרות ברקע (background) בו־זמנית עם שירות ישיר (on־line) למשתמשים.

2. האם המערכת מאלצת או אינה מאלצת את המשתמש להיות מוגבל עקב מגבלותיו של ציוד המערכת.

לא ברור כלל אם יש בכוחן של מערכות שיתוף־זמנים להתחרות מבחינת העלות עם מערכות עיבוד בסדרות עבור אותו סוג תפוקה וקיבולת (capacity), אולם סך הכל של היעילות, כולל יעילות האדם, יכולה לקזז זאת.
יש לזכור, שהעלות עבור השימוש במחשב העובד בשיטת שיתוף־זמנים משקפת את העלות של ציוד ותוכנה הגבוהה במחשבים אלו. אולם עלות הציוד והתוכנה לשיתוף־זמנים הולכים ויורדים.

גורמי תכנון והיבטים כלכליים בשיתוף־זמנים

עיקר הבעיה היא תכנון המערכת למקסימום שימוש ויעילות של אנשים וציוד לגבי העלות הכספית. הגורמים שקובעים בתכנון זה הם עלות כוח־אדם, מחיר ההמתנה לתשובה, מחיר הציוד ומחיר התוכנה.

לפני שנדון בבעיה זו נגדיר מספר מושגים:

משתמש — אדם שנמצא ליד מסוף ונרשם לקבלת שירות.

מספר תחנות — מספר המסופים שדרכם ניתן להתקשר למערכת.

מספר המשתמשים — מספר המשתמשים בכל יחידת זמן.

זמן תגובה — הזמן שחולף מלחיצת מפתח במסוף ועד לקבלת תוצאה משמעותית עקב לחיצת המפתח. סוג התגובה יכול לנוע מחזרה על תו שתוקתק, או יכול להתייחס לזמן החל מבדיקת תוכנית ועד להצגת תוצאותיה של התוכנית למחכה להן. בדרך כלל זמן התגובה אינו מתייחס לזמן ביצוע ריצות יצור.

יעילות המחשב — אותו חלק מזמן המחשב (בדרך כלל זמן יע"מ) שניתן ישירות למשתמש במסוף. זמן המחשב מחולק לשני סוגים — ישיר ועקיף. בזמן ישיר הכוונה לפעולות עבור המשתמש, בזמן עקיף — לכל הפעולות האחרות. הזמן הדרוש למשל, לקביעת תוכן של מונה הנו ישיר; הזמן הנלקח על־ידי המערכת כדי להחליט עם איזה מסוף להתקשר הנו עקיף.

בעזרת הגדרות אלו נוכל לנסח עתה את המטרה בתכנון מערכת שיתוף־זמנים: הבאה למקסימום את מספר "המשתמשים" במסגרת המגבלות של "זמן תגובה" מקסימלי מסוים לכל המשתמשים ו"יעילות מחשב" הגדולה ממינימום מסוים.
כשמדובר במקסימום המשתמשים המקבלים שירות, הכוונה היא גם להשגת מקסימום היעילות שלהם. כאשר המשתמש קשור ישירות (on-line), ההנחה היא שאינו יכול לבצע עבודות אחרות, ובהמתינו ליד המסוף יעילותו אפסית.

מספר המשתמשים המקבלים שירות יעיל הוא מדד להספק (throughput) של המערכת. אם נתעלם מריצות ייצור שאינן דורשות המתנה ליד המסוף, אזי מספר המשתמשים הוא בדיוק המדד ומקסימיזציה שלהם מביאה למקסימום הספק. אופטימיזציה של המערכת אינה בעיה פשוטה כיוון שהמשתנים אינם בלתי־תלויים. בנוסף, משתנים כמו "מספר המשתמשים" ו"זמן תגובה" הנם אקראיים, כך שאנו מדברים בדרך כלל במונחים הסתברותיים, כמו, "70% מזמני התגובה אינם עולים על 5 שניות".

בעיית התזמון (Scheduling)

אחת מבעיות התכנון החשובות במערכת שיתוף־זמנים היא בניית המתזמן (scheduling). זהו אותו חלק ממערכת ההפעלה המחליט כיצד יינתן השירות וכיצד יבוצעו העיבודים. לאופי המתזמן השלכות כלכליות ברורות.

בתכנון שיטת התזמון קיימות מטרות מנוגדות. מצד אחד רצוי שזמן התגובה הממוצע ומספר המשתמשים הממתינים יקטן ככל האפשר. מטרה אחרת היא שהשיטה תכיר בחשיבות המשתמש ובדחיפות השאילתא. בניגוד לזה קיים הרצון לשרת את המשתמשים בצורה הוגנת ולהגביל את זמן ההמתנה. נקודת המוצא בכל תכנון כזה היא שיטה למדידת עלות ההמתנה לתוצאות. לכל משתמש או לכל עבודה יכולה להיות פונקצית־עלות שונה בעוד המטרה הכללית של שיטת התזמוז היא מינימיזציה של סך הכל העלות של כל המשתמשים.

אחת הדרכים הפשוטות לתזמון היא מתן זמן עיבוד שווה לכל משתמש בהגיע תורו (round־robin או time־slicing). יתרונות שיטה זו ביחס ל"שירות ראשון לבא ראשון" ניכרים ככל שגדלה אי־הוודאות ביחס לזמן ולעיבוד הדרוש. במקרים הבלתי שכיחים שזמן העיבוד ידוע אפריורי, אזי שיטה המבוססת על "הטיפול הבא בג׳וב הקצר ביותר" יתן את זמן ההמתנה הקצר ביותר. ככל שבקיאים יותר בשימוש במחשב, כך כדאי יותר לסטות משיטה קלסית של זמן עיבוד שווה.

ניתן לשפר את יעילות השירות כאשר המתזמן פועל על יותר מדרגת עדיפות אחת. בשיטה זו, שאילתות עיבוד שלגביהן ידוע שהן צורכות זמן עיבוד רב, מקבלות דרגת עדיפות נמוכה מהקצרות יותר. השיפור הרב ביותר בביצוע המערכת מושג כאשר התפלגות העבודות נבדקת באופן שוטף ושינויי עדיפות נעשים באופן דינמי.

שיטת התזמון יכולה לנצל גם מידע אפריורי. למשתמש יש בדרך כלל מושג כמה זמן הוא יצרוך. מידע זה ניתן לספק לתהליך התזמון.

ניקח דוגמא אחת לשיטת תזמון:

ניתן לחלק תוכניות במערכת שיתוף־זמנים לארבעה סוגים:

1. עבודות של שיחה (דיאלוג) — בעבודות אלו משוחח המשתמש עם המחשב כמו מנהל עם מומחי מטה. הוא מספק למחשב הוראות, או שואל שאלות

הדורשות מעט חישובים .דוגמא לעבודה כזו היא בקשה לשליפת מידע מסוים מקובץ. האופייני לה הוא כי המשתמש מצפה לתשובה מהירה ביותר. דבר זה אומר שהמערכת חייבת להתקשר למשתמש כזה בתדירות גבוהה. כיוון שכמות החישובים קטנה, הזמן הדרוש בכל שאילתא של המשתמש הוא קצר.

2. עבודות standby — דורשות יותר חישובים והתשובה חייבת לבוא תוך זמן קצר הואיל והמשתמש מחכה לתשובות ליד המסוף. דוגמא לעבודה כזו היא הרצת ניסוי של תוכנית כדי לראות אם היא פועלת.
עבודות מסוג 2, 1 הן בעלות עדיפות גבוהה של עיבוד.

3. עבודות הפקה (ייצור) — העבודה מופקדת עם הוראות כיצד להריצה. היא אינה דורשת ממשתמש לחכות ליד מסוף; מבחינה זו אלו הן עבודות של עיבוד בסדרות. הן יכולות להגיע למרכז החישוב או להישלח ממסופים.

4. עבודות מינהליות — איסוף נתונים על המערכת עצמה, תוכניות הנהלת החשבונות של המחשב.

המתזמן נותן את העדיפות הגבוהה ביותר לעבודות השיחה, עדיפות שנייה לעבודות standby ועדיפות נמוכה לעבודות הפקה. מבוצע תזמון במעגל סגור לכל אחת מהעבודות, אולם במסגרת מערכת רבת־עדיפויות. כמויות זמן המחשב עבור הסוגים השונים נקבעות מראש. למשל, יחידת הזמן לעבודות השיחה יכולה להיות 20 אלפיות שנייה, ויחידת הזמן ל־standby יכולה להיות 200 אלפיות שנייה (לעבודה).

להלן תהליך התזמון בדוגמא זו:
ראשית ניתן שירות לכל עבודות השיחה בצורת מעגל סגור. לאחר מעגל אחד של כל עבודות השיחה, נותנת המערכת שירות בצורת מעגל סגור לעבודות ה־standby. לכל עבודה כזו ניתנת יחידת הזמן שהוקצתה לאותה דרגת עדיפות. בגמר הזמן שהוקצב לעבודה זו חוזר השירות לעבודות השיחה. אם נותר עדיין זמן לפני החזרה לעבודות השיחה, הוא ניתן לעבודות ההפקה.
העבודות המינהליות הן בעדיפות הנמוכה ביותר, או ברקע (background), כלומר, הן מבוצעות רק כאשר אין דרישה לשירות מעבודות בעלות עדיפות גבוהה יותר.
מערכת תזמון כזו בנויה על כך שגובים מהמשתמש דמי שירות בהתאם לזמן ולרמת העדיפות. דבר זה מרתיע את המשתמש מלהרשם בדרגת עדיפות גבוהה יותר מזו שהוא זקוק לה.
דוגמא זו ממחישה שימוש במידע אפריורי על העבודה שהמשתמש יכול לספק למערכת כדי לשפר את יעילותה.

מטבע הדברים שהמחשב בטל לפעמים מעבודה (בהנחה שהביקוש לשירות אקראי), לפיכך צריך לדאוג שיימצאו עבודות בדרגת עדיפות נמוכה לשם מילוי הזמן הפנוי.

מילון מונחים

אנגלי-עברי

A

ACCESS TIME — זמן גישה
ACCUMULATOR — אוגר
ADDER — מחבר
  FULL ADDER — מחבר מלא
  HALF ADDER — מחבר למחצה
ADDRESS — כתובת

ADP — AUTOMATIC DATA PROCESSING — ענ"א — עיבוד נתונים אוטומטי

ALGOL — ALGORITHMIC ORIENTED LANGUAGE

שפת תכנות עילית המותאמת לפתרון בעיות מדעיות

ANALOG COMPUTER — מחשב אנלוגי
AND GATE — שער ו
APPLICATIONS PROGRAMMER — מתכנת יישומים
ARITHMETIC/LOGIC UNIT — יחידה אריתמטית/לוגית
ASCII COPE — צופן המכיל 7 סיביות
ASSEMBLER — שפת תכנות הקרובה לשפת מכונה

ASYNCHRONOUS COMPUTER

מחשב שבו פעולה אחת מתבצעת לאחר סיום פעולה
אחרת, בניגוד לביצוע פעולות המתואם עם לוח זמנים חיצוני.

AUXILIARY (PERIPHERAL) EQUIPMENT

ציוד שאינו מופעל ישירות בתהליך עיבוד הנתונים, כגון,
יחידות קלט/פלט.

AUXILIARY STORAGE — יחידת אחסון חיצונית (זכרון משני)

B

BASIC — שפת תכנות עילית המתאימה במיוחד לתקשורת
BATCH PROCESSING — עיבוד בסדרות
BAUDOT CODE — צופן בו חמש סיביות מייצגות תו אחד

BENCHMARK PROBLEM — חישוב מסוים המשמש למדידת הביצוע של מכונות
או תוכניות שונות

BCD (BINARY CODED DECIMAL) — צופן עשרוני על בסיס בינרי

BINARY CELL — יחידת אחסון המסוגלת לאחסן ספרה בינרית אחת.

BINARY CODE — צופן בינרי

מבנה בינרי	BINARY FORMAT
מחבר למחצה — ראה ADDER	BINARY HALF ADDER
סיבי״ת — (ספרה בינרית)	BIT (BINARY DIGIT)
תרשים מלבני	BLOCK DIAGRAM
הגישה האבולוציונית (מלמטה למעלה)	BOTTON UP
מספר סיביות לשנייה	BPS — BITS PER SECOND
טעות בתוכנית, במערכת, או פגם בציוד	BUG
מאגר	BUFFER
ראה AUXILIARY STORAGE	BULK STORAGE
בית	BYTE

C

מכונת ניקוב כרטיסים	CARD PUNCH
קוראת כרטיסים	CARD READER
ערוץ	CHANNEL
קיבולת ערוץ	CHANNEL CAPACITY
תו	CHARACTER
מיתוג מעגלים	CIRCUIT SWITCHING
שפת תכנות עילית המותאמת לשמושים מסחריים	COBOL
צופן	CODE
תקשורת	COMMUNICATIONS
תוכנית תרגום	COMPILER
מסוף המפעיל	CONSOLE
כרטיס פיקוח	CONTROL CARD
יחידת הבקרה והפיקוח	CONTROL UNIT
זכרון טבעות	CORE MEMORY
יע״מ — יחידת עיבוד מרכזית	CPU — C CENTRAL PROCESSING UNIT
התקן אחסון בגישה ישירה	CRAM — CARD RANDOM ACCESS MEMORY

D

נתונים	DATA
מסד (בסיס) נתונים	DATA BASE
התקן אחסון בגישה ישירה	DATA CELL

DATA COMMUNICATIONS	תקשורת נתונים
DEBUG	תהליך איתור ותיקון שגיאות בתכנות מחשב
DECISION TABLE	טבלת החלטה
DECK	קבוצת כרטיסים מנוקבים
DECODER	מפענח
DEVICE SHARING	שיתוף בציוד
DIGITAL COMPUTER	מחשב ספרתי
DIRECT ACCESS STORAGE	אחסון בגישה ישירה
DISK DRIVE	כונן תקליט מגנטי
DUMP	העתקה או הדפסה של חלק מהתוכן או כל התוכן של יחידת אחסון
DUPLEX	מערכת המאפשרת תקשורת דו־כוונית

E

EDP — ELECTRONIC DATA PROCESSING	עיבוד נתונים אלקטרוני
ENCONDER	מצפין
EXCESS 3 CODE	צופן עודף 3
EXTERNAL STORAGE	אחסון חיצוני

F

FEASIBILITY ANALYSIS (STUDY)	חקר ישימות
FIELD	שדה
FILE	קובץ
FILE ORGANIZATION	ארגון קבצים
FILE SHARING	שיתוף קבצים
FLIP—FLOP	סב־סוב
FLOWCHART	תרשים זרימה
FORTRAN	שפת תכנות עילית מדעית

G

GENERAL PURPOSE COMPUTER	מחשב המשמש לפתרון מיגוון רחב של בעיות (רב־תכליתי)

H

HALF ADDER	מחבר למחצה
HARDWARE	חמרה (ציוד מחשב)
HEXADECIMAL CODE	צופן הקסאדצימלי
HIGH LEVEL LANGUAGE	שפה עילית
HYBRID COMPUTER	מחשב המשלב יכולת אנאלוגית וסיפרתית.

I

I/O — INPUT/OUTPUT	קלט/פלט
INDEX—SEQUENTIAL FILE ORGANIZATION	ארגון קבצים טבלתי
INFORMATION	מידע
INPUT	קלט
INTEGRATED CIRCUIT	מעגל משולב
INTEGRATED INFORMATION SYSTEMS	מערכות מידע משולבות
INTERFACE	מימשק
INTERNAL MEMORY	זיכרון פנימי

J

JOB	מספר תוכניות המהוות יחידת עבודה אחת עבור המחשב

K

K	יחידה מקובלת לציון קיבולת זיכרון מחשב

L

LINE PRINTER	מדפסת
LOOP	לולאה

M

MACHINE LANGUAGE	שפת מכונה
MAGNETIC TAPE	סרט מגנטי
MAIN SITE COMPUTERS	מחשבים מרכזיים ברשת מחשבים
MARK READING	קריאת סימנים אופטית
MASTER FILE	רב קובץ

MESSAGE SWITCHING AND ROUTING
ניתוב הודעות במערכת תקשורת נתונים

MICR — MAGNETIC INK CHARACTER READING
קריאת סימנים מגנטית

MINICOMPUTER מחשב זעיר

MNEMONIC LANGUAGE שפה סימבולית

MULTIPROCESSING עיבוד בו־זמני של מספר תוכניות

MULTIPROGRAMMING אפשרות במחשב לטפל במספר תוכניות בו־זמנית

N

NOISE רעש — גורם המפריע בהעברת מידע

NUMERIC מספרי

O

OCTAL CODE צופן על בסיס 8

OFF LINE ציוד או התקן שאינו פועל בפיקוחה הישיר של יחידת העיבוד המרכזית

ON LINE ציוד או התקן הקשור ישירות ליחידת העיבוד המרכזית

OPERATING SYSTEM מערכת הפעלה

OPTICAL READING קריאה אופטית

OVERFLOW גלישה

P

PACKET SWITCHING ניתוב צרורות

PARALLEL DATA TRANSMISSION שידור נתונים במקביל

PL/1 — PROGRAMMING LANGUAGE ONE שפת תכנות עילית, המבוססת על שילוב של COBOL ו־FORTRAN

PRINTER מדפסת

PRODUCTION SUPPORT סיוע ליצור, סיוע להפקה

PROGRAM תוכנית

PROGRAMMER תוכניתן

PROGRAM SHARING שיתוף בתוכניות

PROGRAMMED DECISIONS החלטות מתוכנתות

PROJECT TEAM	קבוצת הפרויקט
PUNCHED CARD	כרטיס מנוקב
PUNCHED TAPE	סרט ניר מנוקב

R

RANDOM ACCESS STORAGE	זיכרון בגישה אקראית
RANDOM/DIRECT FILE ORGANIZATION	ארגון קבצים אקראי
READER	מתקן המשמש לקריאת נתונים
READ/WRITE HEAD	ראש קריאה/כתיבה
REAL TIME	זמן אמיתי
RECORD	רשומה
REMOTE COMPUTING SYSTEMS	מערכות עיבוד מרוחקות
RESOURCE SHARING	שיתוף משאבים
RJE — REMOTE JOB ENTRY	הזנת עבודות מרחוק

S

SERIAL DATA TRANSMISSION	העברת נתונים בטור
SEQUENTIAL FILE ORGANIZATION	ארגון קבצים סדרתי
SIMPLEX	מערכת תקשורת
SOFTWARE	תוכנה, תוכניות מחשב
SPECIAL PURPOSE COMPUTERS	מחשב המתוכנן לביצוע סוג מסויים של בעיות (חד־תכליתי)
STATEMENT	הוראה, פקודה בתוכנית
STRING	מהרוזת
STRUCTURED DECISION	החלטה מובנית
SUBROUTINE	שיגרה
SYSTEM	מערכת
SYSTEMS ANALYSIS	ניתוח מערכות
SYSTEMS ANALYST	מנתח מערכות
SYSTEM DEVELOPMENT	פיתוח מערכת
SYSTEM ENGINEERING	תיכנון מערכת המידע
SYSTEM PROGRAMMER	תוכניתן מערכת הפעלה

T

TAPE DRIVE	כונן סרט מגנטי
TECHNICAL SUPPORT	סיוע טכני
TELEPROCESSING	עיבוד רחק (מרחוק)
TERMINAL	מסוף
TIME SHARING	שיתוף זמנים
TOP DOWN	גישה מלמעלה למטה
TOTAL INFORMATION SYSTEM	מערכת מידע כוללת
TRACK INDEX	מדריך המסילות

U

UNSTRUCTURED DECISION	החלטה שאינה מובנית
UTILITIES	תוכניות עזר

מחשבים ומערכות־מידע נועד לספק את הרקע העיוני להוראת הנושא "המחשב ופעולתו בארגונים". מטרת הספר להקנות לקורא בסיס להבנת הפעולה, המבנה והשימושים של המחשבים ומערכות־המידע וכן להעמיד לרשותו טרמינולוגיה שתאפשר לו תקשורת טובה יותר עם מומחי המחשבים. אין כאן כוונה להכשיר מומחי מחשבים, מנתחי מערכות, או תוכניתנים, אלא להקנות לסטודנטים ולמנהלים ידע בנושאי המחשב ורכיביו, בתכנות, במערכות־מידע ובהיבטים ניהוליים וכלכליים של מערכות ממוחשבות.

"גומא" ספרי מדע ומחקר

צ'ריקובר מוציאים לאור, בע"מ